UNIVERSIDAD AUTONOMA DE MADRID

ESCUELA POLITECNICA SUPERIOR

Grado en Ingeniería de Tecnologías y Servicios de Telecomunicación

TRABAJO FIN DE GRADO

DESARROLLO DE UN CURSO/TALLER DE INTRODUCCIÓN A TECNOLOGÍAS DE TRATAMIENTO DE IMÁGENES

Rocío Pascual Cozar
Tutor: José María Martínez Sánchez

JUNIO 2019

DESARROLLO DE UN CURSO/TALLER DE INTRODUCCIÓN A TECNOLOGÍAS DE TRATAMIENTO DE IMÁGENES

AUTOR: Rocío Pascual Cozar
TUTOR: José María Martínez Sánchez

Video Processing and Understanding Lab
Dpto. Tecnología Electrónica y de las Comunicaciones
Escuela Politécnica Superior
Universidad Autónoma de Madrid
Junio de 2019

Resumen (castellano)

Este Trabajo Fin de Grado trata de un curso/taller en el que se ha realizado una iniciación al tratamiento de imágenes digitales mediante el uso de Matlab. Se han realizado paralelamente dos cursos con niveles algo distintos para estudiantes de diferentes edades. El curso principal está enfocado a estudiantes preuniversitarios como un curso base, mientras que el segundo, que es una adaptación del primero, está dirigido a alumnos de la ESO y Bachillerato. Ambos enfocados a estudiantes con curiosidad por las tecnologías y que están pensando en estudiar, en un futuro, una ingeniería. En este taller se han introducido algunos de los aspectos fundamentales de la representación y manipulación de imágenes. A través de la herramienta Matlab, y con el material docente realizado en este trabajo de fin de grado, los alumnos aprenden conceptos básicos que rodean al tratamiento de imágenes y pueden entender mejor que hay detrás de las aplicaciones de edición. Después de una breve explicación sobre la plataforma de programación que se ha utilizado (Matlab), con el fin de que el alumno entre en contacto con ella, se realizará una introducción al mundo del procesamiento de imágenes. Mediante el desarrollo de prácticas sencillas aprenden diferentes aspectos del tratamiento de imágenes, como la mejora de una imagen para su visualización, transformaciones de color, la aplicación de diferentes efectos, identificación de objetos o extracción de información de una imagen… bajo la ayuda y supervisión del profesorado del curso. Una vez realizadas las prácticas proporcionadas, se espera que, al final del taller, el estudiante sea capaz de combinar los conocimientos adquiridos en las sesiones anteriores con el fin de crear un filtro deseado. Con este curso aprenden una nueva herramienta de programación dónde pueden seguir desarrollando sus futuras ideas y creaciones. Es un curso adaptable a diversos niveles que ha sido y puede ser utilizado, en proyectos que fomenten los estudios entre jóvenes de las carreras STEM.

Abstract (English)

This Bachelor Thesis is about a course/workshop in which an initiation has been made to the treatment of digital images using Matlab. Two courses have been held in parallel with different levels for students of different ages. The main course is focused on pre-university students as a base course, while the second one, which is an adaption of the first one, is aimed at high school students. Both of them focused on students that are curious about technologies and are thinking about studying, in the future, an engineering. In this workshop, some of the fundamental aspects of the representation and manipulation of images have been introduced. Through the Matlab tool, and with the teaching material done in this Bachelor Thesis, the students learn basic concepts that surround the treatment of images and are able to understand better what is behind the editing applications. After an introduction to the world of image processing, a brief explanation is given about the programming platform that is going to be used (Matlab) for the student to get in touch with it. Through the development of simple practices, they learn different aspects of the treatment of images, such as the improvement of an image for its visualization, color transformations, the application of different effects, identification of objects or extraction of information from an image... under the help and supervision of the course teachers. Once the practices provided are completed, it is expected that, at the end of the workshop, the student is capable of combine the knowledge acquired in the previous sessions in order to create the desired filter. With this course, they learn a new programming tool where they will be able to continue developing their future ideas and creations. It is an adaptable course at different levels that has been and can be used in projects that promote studies in STEM deegres among young people.

Palabras clave (castellano)

Tratamiento de imágenes, digital, curso, imagen, edición.

Keywords (inglés)

Image processing, digital, course, image, edition.

INDICE DE CONTENIDOS

INDICE DE FIGURAS

1 Introducción

1.1 Motivación

El procesamiento digital de imágenes se entiende como el almacenamiento, transmisión, y representación de información de imágenes digitales por medio de una computadora digital. El aspecto principal en el que se basa el interés en el procesamiento digital de imágenes es la mejora de la información que contiene una imagen para la interpretación humana y la extracción de información útil de la misma.
Procesar una imagen incluye los siguientes tres pasos: importación de la imagen a través de herramientas de adquisición de imágenes, análisis y manipulación de la imagen y, por último, la visualización de la imagen modificada.

Esta memoria de TFG abarca el desarrollo de un curso de introducción al tratamiento de imágenes que ha sido desarrollado en dos niveles para edades diferentes. El curso base, del que se ha partido, se enfoca como un taller de iniciación a la universidad (preuniversitario). A partir de él se ha desarrollado un segundo curso de menos nivel, que ha sido utilizado y adaptado para la fase 3 de la jornada de 'Quiero Ser Ingeniera' (QSI) en la Escuela Politécnica Superior de la Universidad Autónoma de Madrid (UAM) [1]. Lo que ha servido como prueba para futuros cursos y mejoras.

Se trata de un Trabajo de Fin de Grado doble donde Lucía Pastor Cendoya y yo hemos desarrollado material para un curso de introducción al tratamiento de imágenes. Cada una ha realizado su curso con su material docente correspondiente por separado, con la posibilidad de unir ambos talleres para proyectos futuros. Posteriormente, este material ha sido usado de manera conjunta para en colaboración con otros generar el material para 'Quiero ser Ingeniera'.

Se espera que el desarrollo de este TFG sirva de ayuda y como introducción, al tratamiento de imágenes, para personas con poco o nulo conocimiento en el tema.

1.2 Objetivos

El objetivo de este TFG es el desarrollo de material docente y prácticas para un curso/taller de iniciación a tecnologías de tratamiento de imágenes. Este curso/taller puede ser adaptado a diversos niveles de los potenciales estudiantes. A partir del material base se pueden desarrollar distintos cursos que sirvan como iniciación al mundo del tratamiento de imágenes. Estos cursos se pueden adaptar tanto en tiempo, como en temario, formato y objetivos, cumpliendo con las necesidades de cada nivel y conocimientos de los estudiantes. Es un curso abierto a posibles cambios y futuras mejoras con el fin de desarrollar más niveles, aparte de los citados en este TFG.

El curso cubre distintos aspectos de la cadena de imagen: adquisición, representación y almacenamiento, procesado básico (e.g., realce, ajuste de brillo, reducción de ruido), procesado medio (e.g., detección de bordes, contornos, identidades de objetos individuales) y avanzado (e.g. filtros tipo Instagram), y presentación. El objetivo final será un proyecto donde se mejoren varias imágenes dañadas y se obtengan las dos imágenes más creativas posibles, imitando algunos efectos de las apps más populares.

[1] https://quieroseringenierauam.es

Con este curso de introducción al tratamiento de imágenes se busca cubrir algunos huecos que existen entre el paso de la ESO y el Bachillerato a los años de universidad. Sería una forma de introducir al alumno/a en el mundo universitario y una manera realista de que conozca las herramientas y los términos que posteriormente va a usar, si se decide por una ingeniería. No se pretende que sea material base para poder crear no un curso nivel universitario, si no pre universidad, curso 0/iniciación o un curso para fomentar las carreras STEM.

1.3 Estado del arte

La información visual es el tipo de información percibida más importante, procesada e interpretada por el cerebro humano. Un tercio del área cortical del cerebro humano está dedicada al procesamiento de información visual.

El procesamiento digital de imágenes, como tecnología basada en computadora, realiza el procesamiento, la manipulación y la interpretación automáticos de dicha información visual, y desempeña un papel cada vez más importante en muchos aspectos de nuestra vida diaria, así como en una amplia variedad de disciplinas y campos. En ciencia y tecnología, con aplicaciones como televisión, fotografía, robótica, teledetección, diagnóstico médico e inspección industrial.

El tratamiento de imágenes ha ido evolucionando a lo largo de los años para ir ofreciendo información más fiable, precisa y detallada. Esto ha estado ligado a la constante evolución que ha ido teniendo la fotografía. Si nos centramos en la edición de imagen vemos que hoy en día existen numerosos programas de edición de imágenes digitales y se encuentran al alcance de la mayoría de las personas.

Con cursos como el desarrollado en este TFG, se quiere conseguir la introducción del estudiante en el mundo del tratamiento de imágenes a través de la edición de imágenes. Para ello se hace uso de la herramienta Matlab que permite procesar imágenes digitales, de proyectores para las explicaciones teóricas y la exhibición de resultados y de webcams con las que se tomarán las imágenes usadas en Matlab.

Es un TFG de carácter pedagógico pues tiene como objeto la educación y el aprendizaje de los estudiantes en temas de procesamiento de imágenes mediante el uso de las herramientas antes expuestas. Para este curso se han tomado como referencia cursos de años pasados realizados también en la Universidad Autónoma de Madrid como son el curso de *Robótica* y el de *Videojuegos* (ver Apéndice A). Debido al buen resultado que obtuvieron por parte de los alumnos, este año han vuelto a participar en las jornadas de ‘Quiero ser Ingeniera’. En ellos, los alumnos recibían unas clases teóricas breves sobre lo que iban a desarrollar para después aplicarlo en la parte práctica. En el taller de Robótica los alumnos construyen y programan su propio robot real y tangible, de manera que el robot interactúe con el exterior de forma autónoma. En el de Videojuegos, mediante la herramienta GDevelop, que no requiere conocimientos técnicos de programación y diseño, desarrollan su propio videojuego con el que podrán jugar. En ambos talleres los alumnos hacen uso de las herramientas necesarias para poder desarrollar de manera satisfactoria el curso. Son realizados durante 5 días, con sesiones de duración entre 1 hora y media y 2 horas. Tres días con una única sesión y dos días con doble sesión. Se estima que son talleres de una duración aproximada de 14 horas y media.

Con este tipo de cursos se consigue la introducción de los alumnos a campos muy complejos (crear robots, programar videojuegos, procesar imágenes) a través de técnicas "sencillas" que ellos mismos pueden realizar sin tener prácticamente conocimiento en el tema. De esta manera los alumnos desarrollan capacidades y demuestran sus habilidades.

También se han podido tomar como referencia las prácticas de asignaturas del grado de 'Ingenieria de Tecnologías y Servicios de Telecomunicación' como son *Tratamiento de Señales Visuales* y *Tecnologías de Video*. En ellas se cubren diversos temas del procesamiento de imágenes y de video. Para este curso se han tratado muchos de las prácticas vistas en esas asignaturas, pero de manera muy simplificada y con un nivel de dificultad mucho menor. En el ámbito pedagógico la realización de estos cursos puede despertar en los alumnos y alumnas la inquietud o el gusto por temas a los que antes no prestaban atención.

1.4 Organización de la memoria

La memoria consta de los siguientes capítulos:

- En el capítulo 2 se realiza un Diseño del proyecto, donde se plantean los tipos de cursos (niveles) contando sus diferencias en las partes teóricas y prácticas, temas y como se han distribuido estos temas en sesiones. Algo a nivel alto con un índice para temas y sesiones.

- A continuación, en el capítulo 3 de 'Desarrollo de contenidos del curso base' se explica tanto el material docente como práctico realizado y utilizado para el curso base de introducción al tratamiento de imágenes. Habrá una sección por cada tema a desarrollar con algunas imágenes de los resultados obtenidos.

- En el capítulo 4 'Validación en un curso real: taller del Campus Engineering Girl de QSI', se recogen las modificaciones y adaptaciones del curso base realizadas para el campus 'Quiero ser Ingeniera' así como los resultados obtenidos.

- En el capítulo 5 de 'Conclusiones y trabajo futuro' se revisa los cursos realizados y la consecución de los objetivos que se querían alcanzar. También se hace referencia a las posibles adaptaciones en un futuro y a la mejora en el número de estudiantes en el mundo de las carreras STEM.

- Para concluir, las 'Referencias' que incluye las referencias bibliográficas de los artículos y libros consultados durante el desarrollo del proyecto.

2 Diseño

El curso desarrollado en este TFG se ha planteado para tener una duración aproximada de 12 horas. Cada sesión, de dos horas, consta de una parte teórica y una parte práctica. La teórica se explicará de manera sencilla mediante la utilización de presentaciones realizadas en Power Point. Estas presentaciones abarcan los distintos temas que se van a tocar en cada práctica. Están dotadas de un gran número de imágenes que ayudarán a la explicación del temario junto con breves descripciones adecuadas a cada nivel.

El entorno de trabajo utilizado para la parte práctica ha sido Matlab, debido a que es una herramienta fácil e intuitiva de manejar y muy usada en algunos grados de ingeniería. Las presentaciones se han realizado en Power Point (.ppt) y los guiones se encuentran en PDF.

2.1 Niveles

Para el curso base, utilizando el material base de este TFG, la teoría será algo compleja con términos y definiciones que se han visto en cursos posteriores a la ESO (gradientes, algoritmos, derivadas de segundo grado, convolución…). Teniendo en cuenta que lo que se quiere es llamar la atención del alumno, la parte teórica estará dotada de contenido visual. En cuanto a la práctica, se proporcionará un guion en PDF junto con el script con ejercicios de Matlab. Este guion constará de una breve introducción al tema seguida de varios ejercicios a realizar junto con preguntas sencillas en alguno de los ejercicios que el alumno tendrá que comentar. La dificultad que tendrá será ir siguiendo un guion en PDF e interpretarlo. Los scripts de Matlab se realizarán con la extensión .m con el objetivo de que se familiaricen de cara a la universidad y tendrán que rellenar los huecos en los que hay XXX. En el caso de este nivel, el código de casi todas las funciones irá implementado en el ejercicio, no se encontrarán aparte. Se realizará cada ejercicio, primero, con imágenes estáticas y, luego, con imágenes tomadas por los estudiantes mediante la webcam.
Este material base ha sido adaptado, para el nivel de ESO/Bachillerato usado en QSI, de la siguiente manera.

Para el nivel de ESO/Bachillerato la teoría será lo más simple posible teniendo en cuenta el conocimiento que pudieran tener los alumnos en esta área. Se darán breves definiciones que ayudarán a entender de manera muy general el concepto que se está tratando, sin adentrarse en detalles técnicos que pueden confundir al estudiante. Seguirá siendo una presentación visual con menos texto que la anterior. La cantidad de temario será la misma, pero eliminando conceptos que aún no han visto en estos cursos. Por otro lado, la parte práctica constará de un script de Matlab con distintos ejercicios sobre la teoría dada. Para la adaptación se ha optado por la extensión .mlx de Matlab. Este tipo de extensión almacena secuencias de comandos en vivo y permiten diferenciar el texto del código. Así al alumno le resultará más fácil y claro a la hora de realizar la práctica. También se puede escribir y compilar código por secciones, lo que facilita la visualización ejercicio por ejercicio (ver Figura 1).

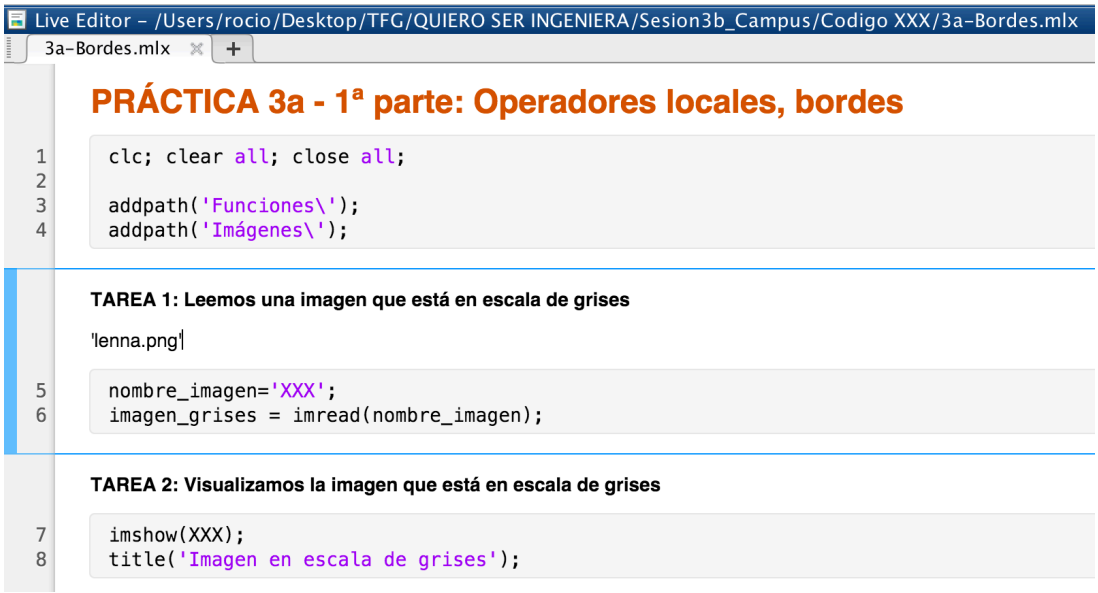


Fig. 1. Live script from Matlab .mlx.

Cada práctica consistirá en la resolución de diversos ejercicios donde tendrán que rellenar partes sencillas del código. Mediante XXX indicamos el hueco a rellenar por el alumno, dando, en ocasiones, alguna pista sobre lo que se debería poner. El grado de dificultad va aumentando a medida que pasamos de práctica, llegando a dejar líneas enteras de código para rellenar por el estudiante. En este nivel (ESO/Bachillerato) no existirá un guion en PDF por práctica, si no que los ejercicios vendrán expuestos al final de cada presentación en forma de imágenes para que el alumno pueda seguir cada paso de manera más sencilla. También se mostrará el resultado que se debe obtener en cada ejercicio acompañado de preguntas sencillas sobre los resultados obtenidos. Se espera así una mayor atención en la tarea propuesta. Por otro lado, cada práctica o tema constará de dos scripts .mlx. En el primer script los ejercicios se realizarán usando imágenes estáticas cogidas de Internet, mientras que en el segundo se usarán imágenes tomadas por los propios alumnos con las webcams facilitadas por la universidad. Consiguiendo una mayor interacción con el estudiante. Para este nivel las funciones que realizan los distintos cambios sobre una imagen se han realizado en scripts aparte modificando sus nombres por unos más intuitivos, dejando el código necesario en el ejercicio. De esta manera los alumnos no tendrán que ver las líneas de código de cada función, que en ocasiones pueden confundirles, y solo verán el nombre de la función que se aplica sobre la imagen.

Por último, para ambos niveles, el proyecto final constará de un script en blanco, con algún comentario como señalización y guía, donde se animará al alumno a utilizar lo visto en las prácticas anteriores para crear distintos efectos sobre una imagen. Contarán con la ayuda de las prácticas anteriores y de una lista de funciones modificadas dada.

Se ha tenido en cuenta que la diferencia de nivel, en el tema de tratamiento de imágenes, entre alumnos de la ESO, de Bachillerato y de inicio a la universidad es poca. Son estudiantes que nunca han trabajado en el entorno de programación de Matlab y que lo único que les puede diferenciar son conocimientos matemáticos.

2.2 Temas

El temario para el curso será el mismo en los dos niveles:

- Introducción al entorno de trabajo (Matlab)

- Introducción a la imagen digital
  - Resolución espacial
  - Cuantificación
  - Imagen Blanco y Negro
  - Imagen RGB

- Transformaciones de color
  - CMY
  - YUV
  - YIQ
  - HSV
  - HSI

- Pasar una imagen a sepia y negativo

- Ajustar brillo, contraste y saturación

- Operadores puntuales
  - Tipos de ruido

- Operadores locales
  - Detección de bordes (Roberts, Prewitt, Sobel y Canny)
  - Suavizado de una imagen con ruido

- Ajustar nitidez

- Efectos
  - Combinación de dos imágenes (suma, resta, multiplicación y división)
  - Montaje de imágenes (al estilo *Warhol*)
  - Efecto *Mirror* (simetría de la imagen)
  - Transformaciones geométricas (traslación, escalado, rotación e inclinación)

- Proyecto final

Los temas realizados por mi han sido: transformaciones de color (CMY, YUV, YIQ, HSV, HSI), pasar una imagen a sepia y negativo, operadores locales (detección de bordes), ajuste de nitidez y efectos (combinación de imágenes, montaje de imágenes, efecto *Mirror* y transformaciones geométricas). El proyecto final ha sido realizado de manera conjunta.

2.3 Sesiones

Partiendo de los temas anteriores, las sesiones para el curso/taller se han dividido de la siguiente manera:

Sesión 1:

- Introducción al entorno de trabajo (Matlab)
- Introducción a la imagen digital
    - Resolución espacial
    - Cuantificación
    - Imagen RGB
    - Imagen blanco y negro

Sesión 2:

- Transformaciones de color.
    - CMY
    - YUV
    - YIQ
    - HSV
    - HSI
- Pasar una imagen RGB a sepia y negativo
- Ajustar brillo, contraste y saturación

Sesión 3:

- Operadores puntuales
    - Tipos de ruido

Sesión 4:

- Operadores locales
    - Detección de bordes (Roberts, Prewitt, Sobel y Canny)
    - Suavizado
- Ajustar nitidez

Sesión 5:

- Efectos
    - Combinación de dos imágenes (suma, resta, multiplicación y división)
    - Montaje de imágenes (al estilo *Warhol*)
    - Efecto *Mirror* (simetría de la imagen)
    - Transformaciones geométricas (traslación, escalado, rotación e inclinación)

Sesión 6:

- Proyecto final

Las sesiones se han ordenado de esta manera teniendo en cuenta lo que se ha ido viendo en prácticas anteriores, para así seguir una cierta coherencia. Se ha considerado que este orden era el indicado para facilitar el entendimiento de los estudiantes. Se puede ver como en una misma sesión se han combinado temas de ambos proyectos desarrollados en paralelo.

3 Desarrollo de contenidos del curso base

A continuación, se presentará un desarrollo de cada uno de los temas realizados por mí para este curso/taller. Cada tema constará de parte teórica y parte práctica. También se expondrán algunos de los resultados obtenidos. En el apéndice B se puede ver un ejemplo de material base docente para el curso de introducción con su presentación, guion y script de Matlab. Para ver más en detalle el material correspondiente a cada sesión ir a la siguiente dirección: http://www-vpu.eps.uam.es/talleres_imagen/ [2]

3.1 Transformaciones de color

El color puede ser definido por su plano rojo(R), verde(G) y azul(B), el conocido sistema RGB, o por sus transformaciones lineales como CMY, YUV, YIQ, entre otras. Estos son los conocidos colores primarios o aditivos (suma de colores de luz). La información captada por una cámara inicialmente es RGB. Esta información puede ser la que nos ofrezca las máximas posibilidades de edición fotográfica, pero esto no quiere decir que sea la imagen que mayor calidad tenga.
Es útil pensar en una imagen en color como una imagen con valor vectorial, donde cada píxel se asocia a ella, como un vector de tres valores. Cada componente de este vector corresponde a un aspecto diferente del color, según el modelo de color que se utilice. Por ejemplo, en un modelo RGB, los tres valores del vector denotan los componentes de color rojo, verde y azul de ese píxel. En un modelo HSV, los tres valores en el vector denotan el tono, la saturación y el valor del color de ese píxel. Y así con el resto de modelos.

En teoría, sobre cualquier modelo de espacio de color se pueden realizar transformaciones de color. Aunque, en la práctica, algunas transformaciones se pueden adaptar mejor a ciertos modelos. Las transformaciones de color se harán sobre la imagen 'india.jpg' dada y posteriormente con la imagen tomada por la webcam (ver Figura 2).

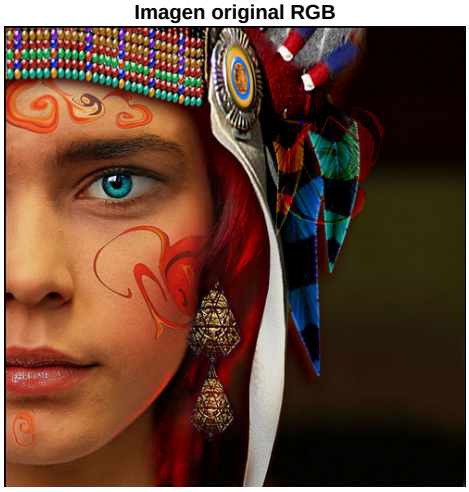


Fig. 2. Imagen india a color.

Antes de comenzar con las transformaciones, como ejercicio inicial se propone intercambiar los canales RGB. Esto es, por ejemplo, cambiar el plano rojo por el verde, el verde por el azul y el azul por el rojo. El 1 corresponde al canal rojo, el 2 al canal verde y el 3 al canal

[2] En caso de problemas de acceso contactar con:
josem.martinez@uam.es

azul. Por lo tanto, el orden inicial de los canales sería: [1 2 3]. Si intercambiamos los canales, el vector quedaría de la siguiente manera: [2 3 1]. Para realizar los siguientes ejercicios habrá que obtener los tres canales RGB por separado de la imagen y después obtener estos 3 canales a color para poder hacer cada una de las transformaciones.

3.1.1 Pasar de RGB a CMY

Los colores secundarios de la luz son el Cian (C), el magenta (M) y el amarillo(Y), estos se denominan sustractivos pues se usan como filtros para sustraer colores de la luz blanca. Cada color está representado por los tres secundarios. Las impresoras y fotocopiadoras en color hacen uso de una entrada CMY o bien de una conversión interna de RGB a CMY. Este sistema es el mismo que RGB, con la diferencia que donde antes había negro ahora existe blanco y viceversa.
Está relacionado con el modelo de color RGB de la siguiente manera:

$$\begin{bmatrix} C \\ M \\ Y \end{bmatrix} = \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} - \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

- Teoría: en la parte de teoría a los alumnos se les contará de manera breve en que consiste el modelo CMY, sus diferencias con el modelo RGB, y como a partir de este último conseguiremos obtener imágenes con los colores secundarios (CMY). Se hará uso de las diapositivas en Power Point para la explicación.

- Práctica: en la práctica realizada para los alumnos el ejercicio propuesto consistirá en obtener los tres canales CMY por separado a partir de los RGB. En este caso la imagen se encuentra entre valores de 0 a 255. En lugar de restarle uno a cada canal RGB, se le restará 255. Al alumno se le pide rellenar las operaciones correspondientes para obtener el Cian, Magenta y Yellow. Por último, visualizarán cada uno de los canales por separado. Los resultados obtenidos serán los siguientes (ver Figura 3).

Fig. 3. Resultado ejercicio CMY.

3.1.2 Pasar de RGB a YUV

Este modelo está compuesto por un canal de luminancia (Y) y otros dos canales de crominancia en los que se codifica color (U y V). La luminancia es la componente de la imagen en blanco y negro con la información del brillo de la imagen. La crominancia es la componente de la imagen con la información del color. El modelo YUV es utilizado en los sistemas de difusión de televisión PAL y NTSC. Es un sistema más similar a la percepción

del ojo humano que el estándar RGB. El ojo humano es más sensible al verde que al rojo o al azul, de los tres colores es el que más ancho de banda ocupa. En la práctica, la componente Y de luminancia tendrá que pasarse a escala de grises mediante *rgb2gray*. La función de transformación que relaciona el modelo YUV con el RGB es la siguiente:

$$Y = 0.299 * R + 0.587 * G + 0.114 * B$$

$$U = 128 - 0.168736 * R - 0.331264 * G + 0.5 * B$$

$$V = 128 + 0.5 * R - 0.418688 * G - 0.081312 * B$$

- Teoría: en la parte de teoría a los alumnos se les dará una breve explicación sobre el modelo YUV, sus características y su relación con el modelo RGB. Se explicarán brevemente los términos de luminancia y crominancia y su relación con los conos y bastones que se encuentran en el ojo humano. Se hará uso de las diapositivas en Power Point para la explicación.

- Práctica: en la práctica realizada para los alumnos el ejercicio propuesto consistirá en obtener los tres canales YUV por separado a partir de los RGB. Esto lo harán mediante la función de transformación dada en teoría, donde tendrán que escribir las distintas funciones para Y, U y V. Por último, visualizarán cada uno de los canales por separado. Los resultados obtenidos serán los siguientes (ver Figura 4).

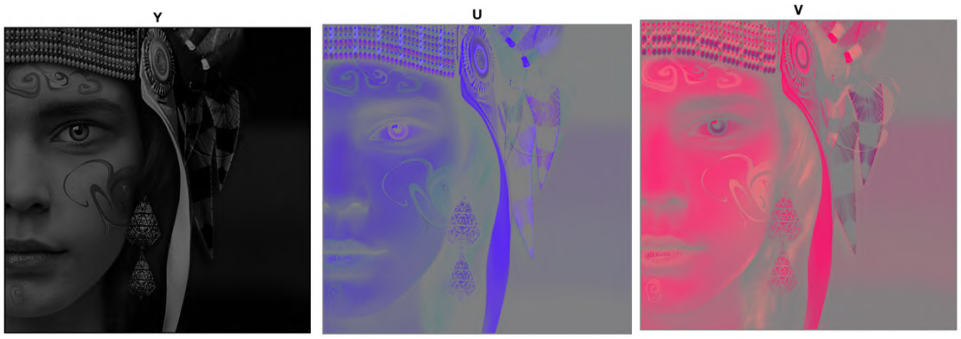


Fig. 4. Resultado ejercicio YUV.

3.1.3 Pasar de RGB a YIQ

Cada color se representa en términos de una componente de luminancia (Y) y dos componentes de crominancia o color: componentes en fase (I) y cuadratura (Q). Este modelo era el usado antiguamente en la transmisión de televisión comercial de Estados Unidos (sistema NTSC). También es un modelo más similar a la percepción del ojo humano. La componente Y proporciona toda la información de video. En el modelo YIQ, las componentes de luminancia y crominancia están desacopladas y se pueden procesar por separado, lo que es una gran ventaja. La componente de luminancia de una imagen puede procesarse sin afectar a su contenido de color. Este sistema está relacionado con el RGB por:

$$Y = 0.299 * R + 0.587 * G + 0.114 * B$$

$$I = 0.595716 * R - 0.274453 * G - 0.321263 * B$$

$$Q = 0.211456 * R - 0.522591 * G + 0.311135 * B$$

Teoría: en la parte de teoría a los alumnos se les dará una breve explicación sobre el modelo YIQ, sus características y su relación con el modelo RGB. Se hará uso de las diapositivas en Power Point para la explicación.

- Práctica: en la práctica realizada para los alumnos el ejercicio propuesto consistirá en obtener los tres canales YIQ por separado a partir de los RGB. Esto lo harán mediante la función de transformación dada en teoría, donde tendrán que escribir las distintas funciones para Y, I y Q. Por último, visualizarán cada uno de los canales por separado. Los resultados obtenidos serán los siguientes (ver Figura 5).

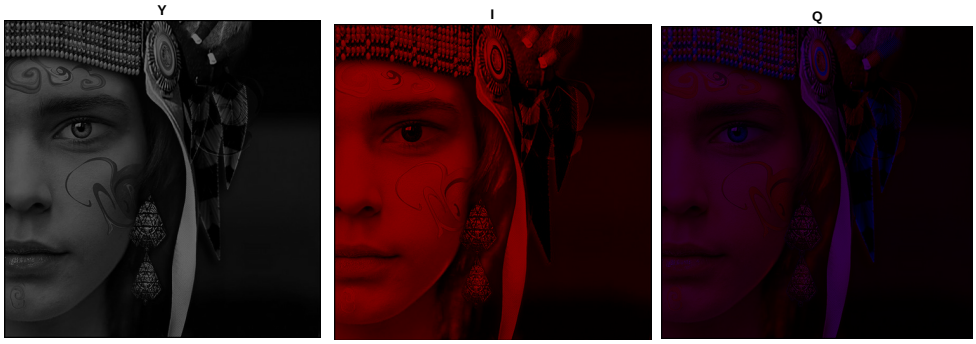


Fig. 5. Resultado ejercicio YIQ.

3.1.4 Pasar de RGB a HSV

Cada color se especifica en términos de Tono (H), Saturación (S) y Valor (V). Las componentes H y S codifican la información de color, mientras que la componente I se corresponde con la intensidad. Las componentes de crominancia (H, S) y valor (V) están desacopladas, lo que supone una gran ventaja para este modelo. El tono y la saturación están íntimamente relacionados con la forma en que el sistema visual humano percibe el color. Este sistema representa el color de una manera mucho más intuitiva. Para el procesamiento de imágenes basado en propiedades del sistema visual humano se utiliza el modelo HSV, ya que el RGB no es demasiado eficiente. Se relaciona con el modelo RGB de la siguiente manera:

$$V = \frac{1}{3}(R+G+B)$$

$$S = 1 - \frac{3}{(R+G+B)}[\min(R,G,B)]$$

$$H = \begin{cases} \theta & B \le G \\ 360 - \theta & B > G \end{cases}$$

$$\theta = \cos^{-1}\left\{\frac{\frac{1}{2}\left[(R-G)+(R-B)\right]}{\left[(R-G)^2+(R-B)(G-B)\right]^{1/2}}\right\}$$

- Teoría: en la parte de teoría a los alumnos se les dará una breve explicación sobre el modelo HSV, sus componentes, características y su relación con el modelo RGB. Se hará uso de las diapositivas en Power Point para la explicación.

- Práctica: en la práctica realizada para los alumnos el ejercicio consistirá en obtener la imagen HSV. Simplemente harán uso de la función de Matlab *rgb2hsv* para realizar la transformación de un modelo a otro, pasándole como parámetro la imagen que quieran transformar. Por último, visualizarán la imagen HSV formada por sus tres componentes. El resultado obtenido es el siguiente (ver Figura 6).

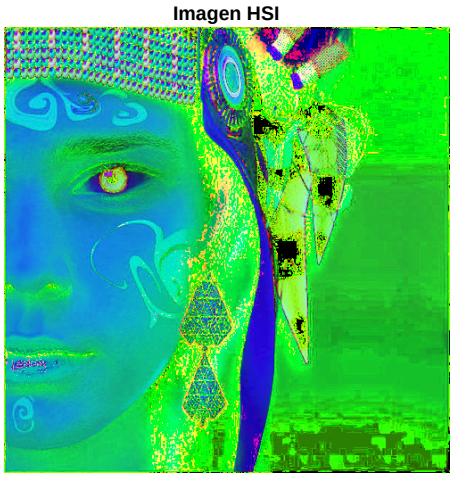


Fig. 6. Resultado ejercicio HSV.

En la parte teórica también se explica de manera breve el modelo HSI. Entre el modelo HSI y HSV apenas hay diferencias a la hora de visualizar la imagen. Se pueden encontrar diferencias en la saturación, pero estas son muy sutiles, por ello se aplica únicamente el modelo HSV.

3.2 Sepia y Negativo

3.2.1 Sepia

Cuando convertimos una imagen RGB a sepia conseguimos un aspecto más amarillento de la imagen. Se reproducen los colores de una foto antigua y produce resultados muy realistas en fotos recientes. Para conseguir este efecto en la imagen bastará con aplicar la siguiente relación con el modelo RGB:

Canal 1= 0.393 * **R** + 0.769 * **G** + 0.189 * **B**

Canal 2 = 0.349 * **R** + 0.686 * **G** + 0.168 * **B**

Canal 3= 0.272 * **R** + 0.534 * **G** + 0.131 * **B**

- Teoría: en la parte de teoría a los alumnos se les dará una breve explicación sobre lo que es el efecto sepia y como se puede conseguir sobre una imagen RGB. Se hará uso de las diapositivas en Power Point para la explicación.

- Práctica: en la práctica realizada para los alumnos el ejercicio consistirá en obtener la imagen en tonos sepia a partir de la función de transformación dada. Tendrán que rellenar las funciones para cada uno de los canales, para después juntarlos y visualizarlos en una misma imagen. El resultado obtenido es el siguiente (ver Figura 7).

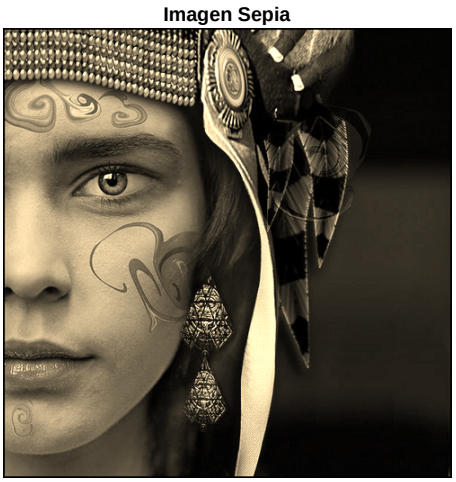


Fig. 7. Resultado ejercicio Sepia.

3.2.2 Negativo

El efecto negativo consiste en invertir los colores de la imagen, el blanco se vuelve negro, el negro se vuelve blanco, el naranja se vuelve azul, etc. Conforme la intensidad de la imagen de entrada aumenta la intensidad de la imagen de salida disminuye.

- Teoría: en la parte de teoría a los alumnos se les dará una breve explicación sobre lo que es el efecto negativo y como se puede conseguir sobre una imagen RGB. Se hará uso de las diapositivas en Power Point para la explicación.

- Práctica: en la práctica realizada para los alumnos el ejercicio consistirá en obtener la imagen en negativo a partir de la función *imcompliment* de Matlab. Tendrán que completar esta función y visualizar la imagen obtenida. El resultado obtenido es el siguiente (ver Figura 8).

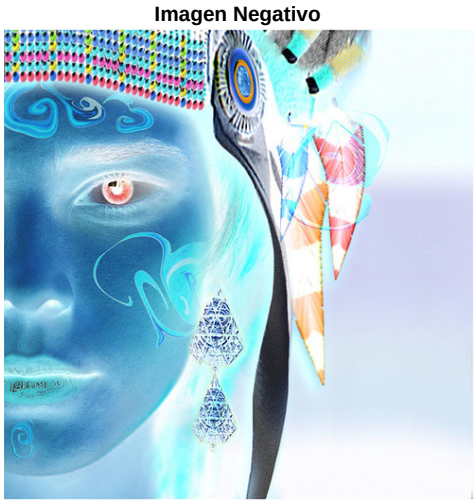


Fig. 8. Resultado ejercicio Negativo.

3.3 Operadores locales

Para detectar los bordes de una imagen hacemos uso de un tipo de operadores denominados locales. Estos operadores están basados en la primera derivada (gradiente) de los niveles de grises en la imagen. A diferencia de los operadores puntuales (explicados en la sesión anterior), en los locales se tiene en cuenta la relación de vecindad entre píxeles. Es decir, cada píxel de salida depende de los píxeles vecinos de entrada. Para la implementación de estos operadores se utilizan máscaras o filtros que al aplicarlos a la imagen entrada nos da la imagen resultado. Los operadores vistos en esta práctica son: Roberts, Prewitt, Sobel y Canny.

Los bordes determinan los cambios de intensidad en el nivel de gris entre dos o más píxeles adyacentes. Pueden aportar una gran información sobre las fronteras de los objetos y también sirven para segmentar imágenes, reconocer objetos y extraer características en regiones de una imagen.

En la parte teórica de esta sesión se explican diferentes conceptos que ayudan a entender mejor los operadores locales: qué son los bordes, qué son los contornos, qué es el ruido, como se aplica una máscara media y para qué sirve un detector de borde.
Debido a que es un tema complejo, del que no tienen conocimiento, se ve muy por encima sin adentrarse en detalles técnicos y destacando sus principales aplicaciones sobre una imagen. Las imágenes para detección de bordes deberán encontrarse en escala de grises. Los siguientes operadores se aplicarán sobre la imagen 'lenna.jpg' y posteriormente sobre la imagen tomada con la webcam (ver Figura 9).

Fig. 9. Imagen lenna.

3.3.1 Roberts

El operador de Roberts es fácil y rápido de computar. Con él se obtiene una buena respuesta ante bordes diagonales. Sin embargo, sus cualidades de detección son pobres, pues es demasiado sensible al ruido. Al ser la máscara utilizada de 2x2, los bordes obtenidos son muy sutiles. Existe una máscara para obtener los bordes verticales y otra para obtener los bordes horizontales:

$$\mathbf{h} = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \text{ y } \mathbf{v} = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix}$$

Bordes horizontales Bordes verticales

- Teoría: en la parte de teoría a los alumnos se les dará una breve explicación sobre cómo se utiliza el operador de Roberts, qué resultado se obtiene con él y las máscaras utilizadas para obtener los bordes verticales y horizontales. Se hará uso de las diapositivas en Power Point para la explicación.

- Práctica: en la práctica realizada para los alumnos el ejercicio consistirá en obtener la imagen de bordes verticales, por un lado, la de bordes horizontales por otro y, en último lugar, la imagen de bordes totales (horizontales y verticales). Tendrán que completar las matrices *v* y *h* para obtener ambos bordes y rellenar los huecos de visualización de imágenes. El resultado obtenido es el siguiente (ver Figura 10).

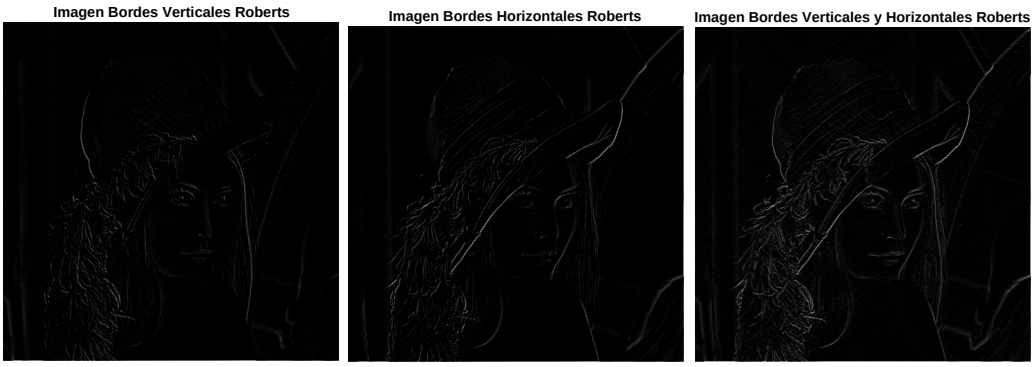


Fig. 10. Resultado operador local Roberts.

3.3.2 Prewitt

En este operador se encuentran involucrados los vecinos de filas/columnas adyacentes, de esta manera ofrecen una mayor inmunidad al ruido. Prewitt detecta los cambios abruptos en la intensidad de la imagen, así como en los límites de la respuesta al ruido. Funciona bien en los casos en los que usamos imágenes con grandes cambios de intensidad y ruidos relativamente bajos. Utiliza máscaras de 3x3, lo que hará obtener mejores resultados que con el operador anterior. Las máscaras para obtener los bordes verticales y horizontales son las siguientes:

$$v = \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix} \qquad h = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix}$$

Bordes verticales Bordes horizontales

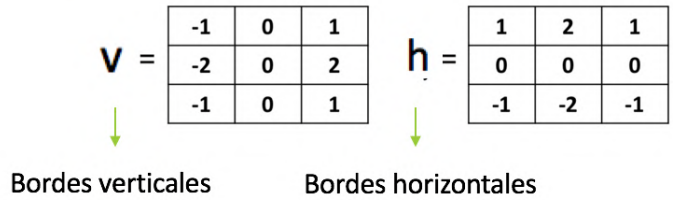
- Teoría: en la parte de teoría a los alumnos se les dará una breve explicación sobre cómo se utiliza el operador de Prewitt, qué resultado se obtiene con él y las máscaras utilizadas para obtener los bordes verticales y horizontales. Se hará uso de las diapositivas en Power Point para la explicación.

- Práctica: en la práctica realizada para los alumnos el ejercicio consistirá en obtener la imagen de bordes verticales, por un lado, la de bordes horizontales por otro y, en último lugar, la imagen de bordes totales (horizontales y verticales). Tendrán que completar las matrices para obtener ambos bordes y rellenar los huecos de visualización de imágenes. El resultado obtenido es el siguiente (ver Figura 11).

Fig. 11. Resultado operador local Prewitt.

3.3.3 Sobel

Este operador es más sensible a los bordes diagonales que el de Prewitt aunque en la práctica hay poca diferencia entre ellos. Sobel es un operador menos sensible al ruido que los dos anteriores y presenta una mayor diferencia de intensidad entre el ruido y los verdaderos bordes. Se obtienen buenos resultados con él, pero aun así no son lo suficientemente buenos para el reconocimiento de ciertas partes de la imagen. Las máscaras para obtener los bordes verticales y horizontales son las siguientes:



- Teoría: en la parte de teoría a los alumnos se les dará una breve explicación sobre cómo se utiliza el operador de Sobel, qué resultado se obtiene con él y las máscaras utilizadas para obtener los bordes verticales y horizontales. Se hará uso de las diapositivas en Power Point para la explicación.

- Práctica: en la práctica realizada para los alumnos el ejercicio consistirá en obtener la imagen de bordes verticales, por un lado, la de bordes horizontales por otro y, en último lugar, la imagen de bordes totales (horizontales y verticales). Tendrán que completar las matrices para obtener ambos bordes y rellenar los huecos de visualización de imágenes. El resultado obtenido es el siguiente (ver Figura 12).

Fig. 12. Resultado operador local Sobel.

3.3.4 Canny

El algoritmo de Canny es el que ofrece mejores resultados para bordes de imágenes con ruido. Este operador rellena los espacios vacíos de la imagen cerrando su contorno siendo el más eficiente y preciso en la detección de bordes. Con este operador, a diferencia de los otros, se pueden encontrar líneas. Presenta buena detección, el algoritmo marca el mayor número real en los bordes de una imagen, una buena localización, los puntos detectados deben estar tan cerca cómo se pueda del centro del borde y respuesta mínima, el borde de una imagen solamente será marcado una vez (el ruido no debe crear falsos bordes).

Se realiza mediante la función *edge* de Matlab y tiene como parámetros de salida th (threshold) y sigma. Si el parámetro de th es bajo se obtienen muchos puntos (generados por el ruido), pero no todos indican contornos. Con un valor alto de th se obtienen menos puntos, pero todos siendo de contorno. Si aumentamos el valor de sigma se reduce el ruido de la imagen, por lo que la veremos con mayor claridad. Se dan intervalos de valores para ambas variables, para límite el intervalo será entre 0 y 1, mientras que para sigma estará entre 0 y 20.

- Teoría: en la parte de teoría a los alumnos se les dará una breve explicación sobre el operador de Canny, sus características y qué resultado se obtiene con él en comparación al resto de operadores. Se hará uso de las diapositivas en Power Point para la explicación.

- Práctica: en la práctica realizada para los alumnos el ejercicio consistirá en obtener la imagen de bordes totales mediante la función *edge* y tendrán que probar con diferentes valores para las variables de límite (threshold) y sigma. Observarán las imágenes obtenidas y verán con qué valores se obtienen mejores resultados de detección. El resultado obtenido para límite 0.1 y sigma 1 es el siguiente (ver Figura 13).

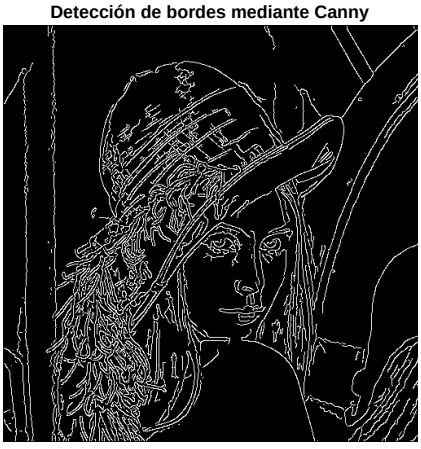


Fig. 13. Resultado operador local Canny.

3.4 Ajustar Nitidez

La nitidez se podría definir como el contraste entre diferentes colores. En el caso de una rápida transición de negro a blanco esta se vería nítida. Si ahora la transición es gradual de negro a gris a blanco esta parecerá borrosa. La técnica de enmascaramiento de nitidez consiste en que una imagen se afila restando una versión borrosa (sin nitidez) de la imagen

de sí misma. Es una manera de realzar los bordes de una imagen o perfilarlos.
Para ello se hará uso de la función *imsharpen* de Matlab, donde mediante los parámetros de Radius (R) y Amount (A) obtendremos una imagen más o menos nítida. El valor de Radius controla el tamaño de la región alrededor de los píxeles del borde que se ven afectados por el afilado. Un valor grande de R afila regiones más anchas alrededor de los bordes, mientras que un valor pequeño afila regiones más estrechas alrededor de bordes. Por otro lado, el valor de A se podría definir como la fuerza del efecto de afilado. Un valor más alto conduce a un aumento más grande en el contraste de los píxeles afilados. Los valores muy grandes para este parámetro pueden crear efectos no deseados en la imagen de salida.
Para la variable R se da un intervalo entre 0 y 10 y para la variable A entre 0 y 2.

- Teoría: en la parte de teoría a los alumnos se les dará una breve explicación sobre el término nitidez, sus características y qué resultado se obtiene con esta función. Se hará uso de las diapositivas en Power Point para la explicación.

- Práctica: en la práctica realizada para los alumnos el ejercicio consistirá en obtener la imagen nitida mediante la función *im_sharpen* y tendrán que probar con diferentes valores para las variables de R (Radius) y A (Amount). Observarán las imágenes obtenidas y verán con qué valores se obtienen mejores resultados de nitidez. El resultado obtenido para Radius 1 y Amount 2 es el siguiente (ver Figura 14).

Fig. 14. Resultado imagen nítida.

Se propondrá un segundo ejercicio de este apartado, en el que se sumará la imagen original a la imagen de bordes totales obtenida como otra forma de perfilar los bordes de una imagen.

3.5 Efectos

Sobre una imagen se pueden aplicar múltiples efectos que hacen que la imagen sea más visual, creativa u original. Se trata de efectos sencillos y llamativos que se pueden combinar y permiten al estudiante jugar con ellos. Algunos de los posibles efectos son los siguientes:

3.5.1 Combinación de imágenes

La combinación de imágenes consiste en la utilización de dos o más imágenes de entrada (A y B) con el fin de producir una imagen resultado de salida (R). Estas imágenes deben tener el mismo tamaño para que sea posible combinarlas. Entre las posibles operaciones aritméticas que podemos realizar se encuentran la suma, la resta, el producto y la división.

SUMA: consiste en la mezcla de dos imágenes => (A+B)/2
RESTA: consiste en obtener la diferencia entre imágenes => ((A-B)/2)+128
PRODUCTO: es un efecto similar al de la suma => (AxB)/255
DIVISIÓN: es la operación contraria a la multiplicación => (A/B)x255

En la imagen resultado los valores de píxel que se obtienen son los valores de píxel de la primera imagen sumada/restada/multiplicada/dividida a los valores de píxel correspondientes de la segunda imagen. Se obtienen buenos resultados con todas las operaciones anteriores menos con el producto, ya que al multiplicar una imagen por otra los valores de salida van a ser demasiado altos y la imagen probablemente se vea muy oscura.

- Teoría: en la parte de teoría a los alumnos se les dará una breve explicación sobre el la combinación de imágenes y sus posibles aplicaciones. Se hará uso de las diapositivas en Power Point para la explicación.

- Práctica: en la práctica realizada para los alumnos el ejercicio consistirá en obtener la imagen resultado combinada mediante las distintas operaciones aritméticas. El alumno deberá aplicar la operación deseada y visualizar las imágenes utilizadas y la imagen resultado. El resultado obtenido para la suma es el siguiente (ver Figura 15).

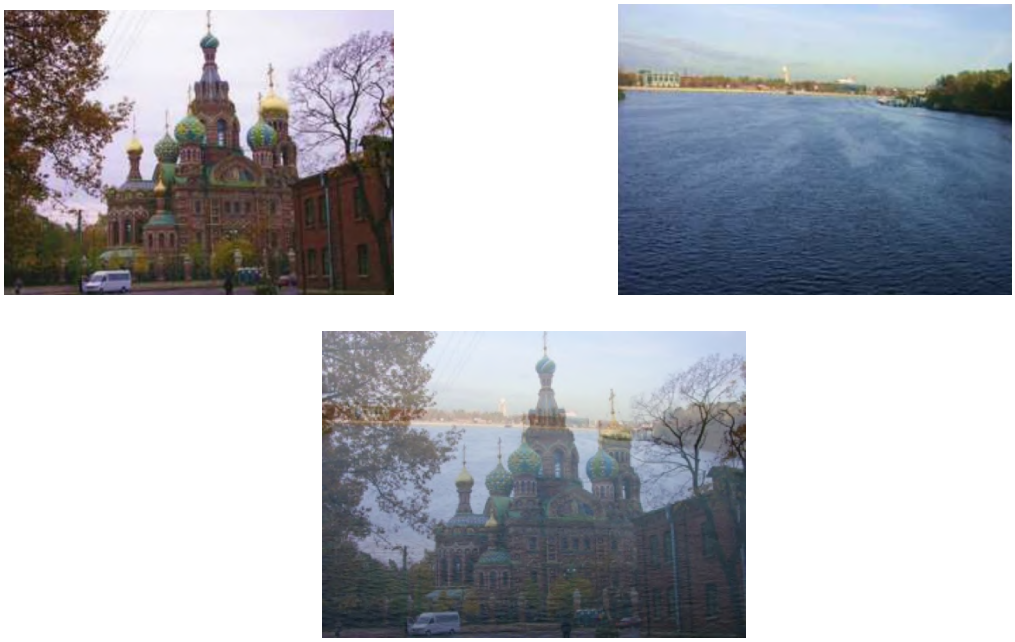


Fig. 15. Combinación suma.

3.5.2 Montaje de imágenes

Otro posible efecto muy llamativo es el montaje de varias imágenes al estilo 'Warhol'. Este montaje se realizará mediante la función *montage* de Matlab que nos permitirá visualizar las imágenes deseadas en una misma imagen, este caso cuatro imágenes (2x2).
Se representará una misma imagen cuatro veces con distintos valores de contraste cada una, es decir se obtendrán cuatro imágenes con contrastes distintos. Esto se realizará mediante la función *imadjust* de Matlab y sus valores tendrán que estar entre 0 y 1, siendo el primer valor más bajo que el segundo [low_in high_in]. Estos valores serán los límites de contraste para la imagen de entrada. Los de salida, por defecto, serán [0 1].

- Teoría: en la parte de teoría a los alumnos se les dará una breve explicación sobre la función *montage* y la función *imadjust* de Matlab. Se hará uso de las diapositivas en Power Point para la explicación.

- Práctica: en la práctica realizada para los alumnos el ejercicio consistirá en obtener cuatro imágenes distintas a partir de una con valores de contraste distintos para cada una de ellas. Tratará de que los alumnos jueguen con los valores de [low_in high_in] para las cuatro imágenes y sean todas de tonalidades diferentes. El resultado obtenido para los valores [0.1 0.5] para la imagen 1, [0.2 0.6] para la imagen 2, [0.3 0.7] para la imagen 3 y [0.4 0.8] para la imagen 4 es el siguiente (ver Figura 16).

Fig. 16. Montaje Warhol.

3.5.3 Efecto Mirror

Este efecto consiste en cambiar la simetría de una imagen. Esta simetría puede ser vertical u horizontal y se realiza mediante la función *flip* de Matlab. Si a esta función se le pasa como parámetro dimensión un 1 se les dará la vuelta a las filas de la matriz de la imagen original. Si ahora se le pasa un 2, estaremos dando la vuelta a las columnas de la matriz de la imagen original.

- Teoría: en la parte de teoría a los alumnos se les dará una breve explicación sobre la función *flip* y como se puede jugar con la simetría de la imagen. Se hará uso de las diapositivas en Power Point para la explicación.

- Práctica: en la práctica realizada para los alumnos el ejercicio consistirá en jugar con la simetría de la imagen cambiando el parámetro dimensión y visualizando la imagen girada. El resultado obtenido para dimensión 1 y para dimensión 2 es el siguiente (ver Figura 17).

Fig. 17. Efecto Mirror.

3.5.4 Transformaciones geométricas

Las transformaciones geométricas modifican la relación espacial entre píxeles. Es una operación que permite encontrar o construir una nueva imagen a partir de una dada. Cada píxel de salida depende de uno de entrada. Las transformaciones afines pertenecen a las geométricas y son las más usadas en imágenes digitales 2D. Una transformación afín es aquella transformación en la que las coordenadas (x', y') del punto imagen son expresadas linealmente en términos de las del punto original (x, y). Entre las transformaciones afines podemos destacar la traslación, la escala, la rotación y la inclinación.

TRASLADAR: consiste en trasladar la imagen original de un punto a otro. Se realiza mediante la función *imtranslate* de Matlab. Esta función le pasa un vector [x, y] como parámetro. Cambiando los valores de x e y podremos mover la imagen en todas direcciones (arriba, abajo, izquierda, derecha). Pueden ser valores positivos y negativos y el rango dado es entre 0 y 100. Si el valor de x es positivo la imagen se traslada a la derecha y se el valor de y es negativo la imagen se traslada hacia arriba.

ESCALAR: consiste en reducir o agrandar la imagen original. Se realiza mediante la función *imresize* de Matlab. Cambiando el parámetro de escala conseguiremos que la imagen se haga más grande o más pequeña. Si el valor de escala es igual a 1 la imagen se queda igual, si es entre 0 y 1 la imagen se reduce y si es mayor que 1 la imagen se agranda.

ROTAR: una imagen puede ser rotada hacia la izquierda o hacia la derecha en función del ángulo. Se realiza a través de la función *imrotate* de Matlab. Cambiando el valor del ángulo que tiene como parámetro giraremos la imagen hacia un lado u otro. Si el ángulo es positivo la imagen girará en el sentido contrario a las agujas del reloj, si es negativo girará en el sentido de las agujas del reloj. Este ángulo puede estar entre 0 y 360 grados.

INCLINAR: consiste en cambiar la perspectiva de la imagen original. Se realiza mediante la función *imtransform* (una transformación espacial en 2D). Con maketform se crea una estructura de transformación espacial que sirve para realizar una transformación N-dimensional afín. Cambiando los parámetros (1) y (2) se conseguirá el efecto deseado. El rango dado para estos valores será entre 0 y 20 y pueden ser positivos o negativos.

- Teoría: en la parte de teoría a los alumnos se les dará una breve explicación sobre las transformaciones geométricas y sus posibles aplicaciones. Se hará uso de las diapositivas en Power Point para la explicación.

- Práctica: en la práctica realizada para los alumnos el ejercicio consistirá en jugar con los distintos parámetros de los que se ha hablado anteriormente y ver el efecto que producen sobre la imagen. Los resultados obtenidos para [20, -10] en la imagen trasladada, para una escala 0.5 en la imagen escalada, para un ángulo de 50 en la

imagen rotada y para unos parámetros -5 (1) y -10 (2) en la imagen inclinada son los siguientes (ver Figura 18).

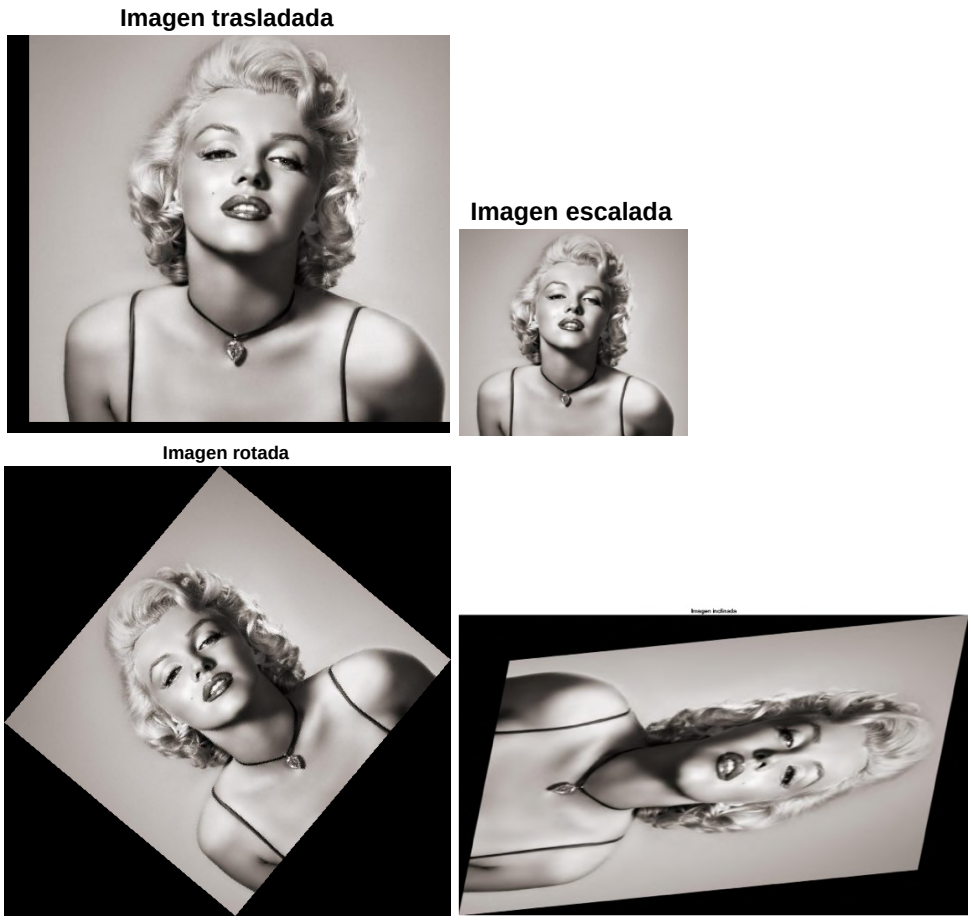


Fig. 18. Transformaciones geométricas.

3.6 Proyecto final

En el proyecto final el objetivo es que el alumno sea capaz de combinar y utilizar los conocimientos aprendidos en las anteriores prácticas con el fin de realizar los cuatro ejercicios propuestos. Los resultados obtenidos se mostrarán frente al resto de alumnos, profesorado y padres.

- Teoría: en la parte de teoría a los alumnos se les dará una breve explicación sobre los ejercicios propuestos para el proyecto final. Se hará uso de las diapositivas en Power Point para su explicación.

- Práctica: la práctica final constará de dos partes. Una primera parte donde al alumno se le entregarán dos imágenes en mal estado con el fin de que las mejoren, en la medida de lo posible, con lo aprendido en los días anteriores. Una segunda parte donde los alumnos tendrán que crear las dos imágenes más creativas combinando partes de otras prácticas. Los alumnos podrán hacer uso de las prácticas realizadas anteriormente y de una lista de funciones modificadas. Los scripts se encontrarán libres de código, simplemente con alguna señalización en texto para guiarles, partirán de cero. La segunda parte se podrá realizar tanto con imágenes de internet como con imágenes tomadas por los alumnos con las webcams.

4 Validación en un curso real: taller del 'Campus Engineering Girl de QSI'

Como ya se ha contado previamente este curso/taller base de introducción al tratamiento de imágenes ha sido adaptado para la fase 3 del campus 'Quiero ser Ingeniera' realizado durante la segunda semana de junio de 2019. El campus ha sido llevado a cabo en la Escuela Politécnica Superior de la Universidad Autónoma de Madrid para alumnas entre 15 y 16 años de edad de diferentes colegios de la Comunidad de Madrid. El material desarrollado para este curso se encuentra en: http://www-vpu.eps.uam.es/talleres_imagen/ [3].

El proyecto ha estado enfocado a la promoción de los estudios de ingeniería entre chicas de cuarto de la ESO, debido a que menos del 25% de los estudiantes universitarios de ingeniería son mujeres. A pesar de que las mujeres son mayoría en las universidades, el porcentaje desciende significativamente cuando se trata de las carreras STEM (Science, Technology, Engineering and Mathematics) (ver Figura 19). Con 'QSI' lo que se pretende es romper con los estereotipos, dar una visión más equitativa y socialmente justa del talento femenino y crear oportunidades para participar en las carreras del futuro y, fundamentalmente, fomentar las vocaciones en grados de tecnología en chicas preuniversitarias.

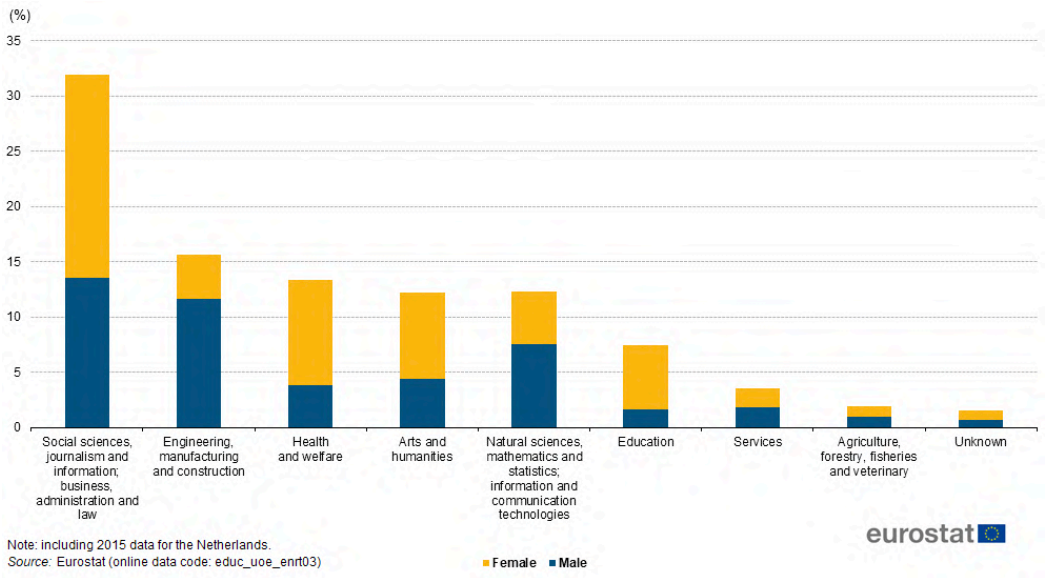


Fig. 19. Distribution of tertiary education students by field and sex, EU-28, 2016.

Gracias al Instituto de la Mujer se ha podido aplicar el curso realizado en este TFG y se han podido obtener valoraciones en las encuestas por parte de las alumnas apuntadas. El taller ha sido adaptado en tiempo, temario, formato y objetivos para el campus, debido a que en un principio el material desarrollado estaba enfocado a un curso de introducción para edades superiores. Por ello, se decidió desarrollar los dos cursos de manera paralela, obteniendo así dos posibles niveles.

Para el campus, el taller debía durar 14 horas y media a lo largo de 5 días. Las sesiones por día fueron de hora y media o 2 horas, realizando tres días una única sesión y los otros dos días doble sesión. En el Apéndice C podemos encontrar el programa del campus QSI. Durante cada sesión el tiempo empleado para la parte teórica fue de 15 minutos a media

[3]En caso de problemas de acceso contactar con: josem.martinez@uam.es

hora, dejando el resto del tiempo para la parte práctica. Contamos con el número de 36 alumnas, que fueron agrupadas en 18 parejas para realizar cada práctica.

En cuanto al temario, este se adaptó tanto en la parte teórica como en la práctica (ver Apéndice D). El formato de diapositivas es distinto al utilizado en el preuniversitario. Estas son mucho más visuales con poco texto y teniendo en cuenta el conocimiento de las alumnas en conceptos matemáticos y tecnicismos. Se eliminaron términos de las diapositivas como gradientes, matrices, algoritmos y derivadas. Las definiciones se redujeron de tal manera que no ocupasen más de dos líneas y fuesen lo más claras y sencillas posibles. En ellas se iba siguiendo cada ejercicio con su resultado a medida que la alumna iba avanzando en la práctica. De tal manera que les resultase mucho más fácil seguir la sesión.

También se adaptó el formato de Matlab y en lugar de usarse la extensión .m se utilizó la extensión .mlx (ver Apéndice E). Se ha considerado que este último formato es más intuitivo y fácil para estudiantes de esta edad. Debido a que este curso tenía dos horas y media más de duración que el que se había planteado (curso base), se añadieron algunos ejercicios adicionales, relacionados con la práctica, en los temas de operadores puntuales, operadores locales y efectos, donde partían de cero en un script de Matlab. En él tenían que seguir las indicaciones de cada ejercicio expuestas en la presentación. También se dejó media hora en la última sesión para realizar las encuestas individuales.

La diferencia principal en comparación al temario del curso base, en el código de los ejercicios, ha sido que las funciones se encontraban en scripts aparte y eran implementadas dentro de funciones con nombres más intuitivos. Así se ha simplificado al máximo el código que las alumnas podían ver y manipular evitando errores y confusiones. Las funciones modificadas han sido las siguientes:

- Transformaciones de color
    - cat => juntar_canales
- Sepia y Negativo
    - imcompliment => RGB_Negativo
- Operadores locales (detección de bordes)
    - imfilter => bordes
    - edge => canny
- Ajustar nitidez
    - imsharpen => dar_nitidez
- Efectos
    - flip => girar
    - cat => juntar
    - imtranslate => trasladar
    - imresize => escalar
    - imrotate => rotar

- imtransform => inclinar

Las sesiones se distribuyeron de la misma manera que se ha indicado en el capítulo 2.

- Sesión 1 (Lunes): Introducción a Matlab y a la imagen digital
- Sesión 2 (Martes): Transformaciones de color, sepia y negativo y ajuste de brillo, contraste y saturación
- Sesión 3 (Miércoles): Operadores puntuales
- Sesión 4 (Jueves – Mañana): Operadores locales y ajuste de nitidez
- Sesión 5 (Jueves - Tarde): Efectos
- Sesión 6 (Viernes – Mañana): Realización proyecto final
- Exposición final (Viernes – Tarde)

Para el proyecto final, aparte de las prácticas anteriores, también se les facilitó una lista con las posibles funciones que podían utilizar para mejorar las imágenes y realizar las creativas. Se les pidió ir comentando en un Word todos los cambios, modificaciones y aplicaciones que iban realizando sobre las imágenes para luego imprimirlo y poder tenerlo como apoyo a la explicación en la exposición final.

Por último, los objetivos que se querían conseguir eran que la alumna se familiarizase con un entorno de programación completamente nuevo (Matlab), que se iniciase en el mundo del tratamiento de imágenes de manera sencilla y que en el proyecto final fuese capaz de utilizar los conocimientos aprendidos para realizar sobre una imagen los cambios deseados. Los resultados fueron bastante buenos, pues las estudiantes a mitad de semana ya se manejaban con soltura por esta nueva herramienta llegando a sus propias conclusiones cuando obtenían un error y realizando las prácticas más rápido de lo previsto. En el proyecto final todas consiguieron mejorar las imágenes propuestas y algunas parejas solo tuvieron tiempo de realizar una imagen. Por otro lado, también he participado en la impartición del taller durante la semana de ‘Quiero ser Ingeniera’ junto con otras compañeras de la escuela.

En la última sesión del viernes, cada pareja de alumnas mostró los resultados obtenidos en el proyecto final frente al resto de alumnos, profesorado y padres en la sala de actos de la universidad mediante un proyector (ver Figura 20).

Fig. 20. Alumnas mostrando el proyecto final.

5 Conclusiones y trabajo futuro

5.1 Conclusiones

El curso que abarca este TFG está compuesto de material base docente propuesto para un curso de introducción enfocado a estudiantes con cierto interés en las TIC. Este taller abarca diferentes aspectos del tratamiento de imágenes. En él se han desarrollado temas que pueden ser llamativos para un estudiante interesado en la edición de imágenes. Las sesiones no se han ordenado de menos a más dificultad, sino que se ha seguido un orden coherente de acuerdo a lo anterior visto. En esta memoria se presentan dos niveles diferentes, pero como se puede observar es un curso adaptable a casi cualquier nivel. Es un taller con una progresión futura.

El curso base desarrollado ha sido adaptado para el Campus QSI y validado, por lo tanto, en un caso real. Es un curso enfocado a alumnas de 15 y 16 años de edad por lo que se han tenido que realizar ciertas modificaciones de tiempo, temario y objetivos, como se ha explicado anteriormente. En el curso adaptado en este TFG se ha comprobado que los resultados y el feedback obtenido por las estudiantes han sido positivos gracias a las encuestas realizadas por las alumnas. Se ha observado que la dificultad del curso ha sido la indicada para estudiantes de esa edad y que han podido realizar las prácticas de la manera que se esperaba. Los principales objetivos se han cumplido y el campus ha sido un éxito, tanto en este taller como en los otros dos ofrecidos (robótica y videojuegos).

5.2 Trabajo futuro

El proyecto ‘Quiero ser Ingeniera’ nos permite probar/experimentar/documentar una posible adaptación futura para futuros campus relacionados con la ingeniería o el tratamiento de imágenes. Se podría adaptar y mejorar a otros campus en base a las encuestas y experiencias obtenidas durante la jornada QSI.

Este curso/taller podría ser adaptado, también, como prácticas preuniversitarias o como curso de iniciación de alguna ingeniería donde se trate el procesamiento de imágenes. Para ello, el nivel de teoría y de prácticas sería más o menos el mismo que en el curso de introducción realizado, pero se tendría que cambiar la dinámica de las sesiones debido a que se trataría de unas prácticas y no de un campus. Si fuese necesario también se podría aumentar o disminuir el tiempo de cada sesión incluyendo o quitando ejercicios de las prácticas, como se hizo en ‘Quiero ser Ingeniera’. En definitiva, este TFG podría adaptarse a múltiples niveles aumentando o disminuyendo el nivel de cada una de sus partes.

Por otra parte, y en paralelo al TFG se espera que como trabajo y objetivo futuro el número de estudiantes chicas en las carreras STEM aumente de manera considerable durante los próximos años. Teniendo como objetivo llegar a un 50% de representación en este tipo de carreras en las universidades. Con más proyectos futuros como ‘Quiero ser Ingeniera’ se espera que se consiga una mayor motivación entre las alumnas y entre sus ideas de estudio también esté presente una ingeniera. También, con jornadas de iniciación a la universidad, podrían aplicarse este tipo de cursos intentando despertar el interés en las TIC en estudiantes de ambos géneros.

Referencias

[1] Alejandro Domínguez Torres, “Procesamiento digital de imágenes”, Perfiles Educativos, num. 72, abril-junio, 1996, Instituto de Investigaciones sobre la Universidad y la Educación

[2] Dra. Nora La serna Palomino, Lic. Ulises Román Concha, Técnicas de Segmentación en Procesamiento Digital de Imágenes

[3] Gonzalez RC, Woods RE 1996. Tratamiento digital de imágenes, Addison-Wesley Publishing Co, Reading, Washington.

[4] Jähne B. 1997. Digital Image Processing, Springer, 4th Edition.

[5] Dr. Rubén Wainschenker, Procesamiento Digital De Imágenes, Clase Teórico Práctica Nº 1, Optativa Área Procesamiento de Señales, Primer cuatrimestre de 2011

[6] John C. Russ, The image Processing – Handbook. Third Edition

[7] Página oficial de Matlab, Mathworks https://es.mathworks.com

[8] “Extensible Markup Language (XML) 1.0 (Second Edition)”, W3C Recommendation 6 October 2000: http://www.w3.org/TR/REC-xml

[9] Castleman KR 1996. “Digital Image Processing”, Prentice-Hall, Englewood Cliffs, New Jersey 07632.

Apéndices

A Talleres Robótica y videojuegos

Taller de Robótica

El Grupo de Robótica de la Escuela Politécnica Superior ha desarrollado talleres de robótica similares en numerosas ocasiones anteriores. Estos talleres se han caracterizado por el notable éxito de participación y la alta calidad de resultados, como lo demuestra la fotografía de algunos de los robots construidos por alumnas que participaron en el concurso que clausura el taller.

Taller de Programación de Videojuegos

El mundo de los videojuegos, gracias a sus facetas colaborativas y el gran atractivo de los entornos 2D y 3D, generan un elevado potencial para utilizar la creatividad y potenciar capacidades. Algunas de las plataformas de creación de juegos, permiten crear un juego completo de forma sencilla y fácilmente exportable, pudiendo elegir desde el aspecto de los personajes del juego, los mapas, hasta el diseño y construcción de cualquier tipo de objeto dentro del videojuego.

En este curso se hará uso de la herramienta GDevelop para la creación de videojuegos que no requiere conocimientos técnicos de programación y diseño. Mediante ella, las alumnas aprenderán los conceptos básicos que rodean a la creación de un videojuego, y potenciarán su creatividad desarrollando su propiojuego desde cero. Después de una introducción al mundo de los videojuegos, se introducirá la plataforma de desarrollo que se va a utilizar, y mediante sesiones muy interactivas, las alumnas aprenderán de forma incremental diferentes aspectos del desarrollo de videojuegos, como la creación y animación de personajes, escenarios, la definición de la lógica del juego e incluso la introducción de métodos de inteligencia artificial que gobiernen el comportamiento de los personajes del juego. Una vez impartidos los conocimientos necesarios para el desarrollo de un videojuego sencillo, cada pareja de alumnas desarrollará su propio videojuego, bajo la ayuda y supervisión del profesorado del curso.

Al final del curso, todas las alumnas habrán desarrollado un videojuego que se podrá ver, jugar y compartir. Se espera que tras la finalización de este curso las alumnas dispongan de las habilidades necesarias para seguir desarrollando sus propios videojuegos, de forma que los conocimientos aprendidos sean de mucha utilidad para sus futuras ideas y creaciones.

B Material base docente

- Las diapositivas utilizadas para el tema de efectos en el curso base de introducción al tratamiento de imágenes son las siguientes:

Curso/Taller Tratamiento de Imágenes

EFECTOS SOBRE IMÁGENES

Rocío Pascual Cozar
rocio.pascualc@estudiante.uam.es

Universidad Autónoma de madrid
E28049 Madrid (SPAIN)

Video Processing and Understanding Lab
Grupo de Tratamiento e Interpretación de Vídeo

Contenido

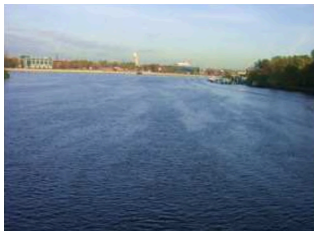
- Combinación de imágenes
  - *Suma*
  - *Resta*
  - *Multiplicación*
  - *División*
- Montaje de imágenes
- Efecto Mirror
- Transformaciones geométricas
  - *Trasladar*
  - *Escalar*
  - *Rotar*
  - *Inclinar*

- **Combinación de imágenes**
  - *Suma*
  - *Resta*
  - *Multiplicación*
  - *División*
- Montaje de imágenes
- Efecto Mirror
- Transformaciones geométricas
  - *Trasladar*
  - *Escalar*
  - *Rotar*
  - *Inclinar*

Combinación de imágenes

- Combinación de imágenes: utilizar dos o más imágenes de entrada para producir una imagen de salida.
  - Entrada: imágenes A y B.
  - Salida: imagen R.
- Posibles operaciones aritméticas de combinación:
  – Aritméticas: suma, resta, producto/división, media
- En principio, todas las imágenes deben ser del mismo tamaño.

+
-
x
/

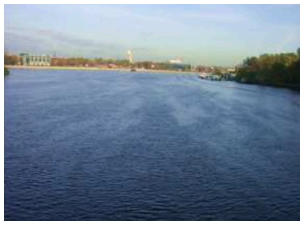
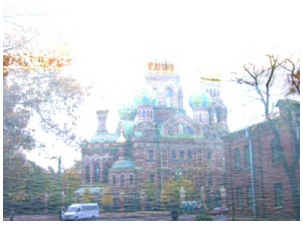


= R

Combinación de imágenes

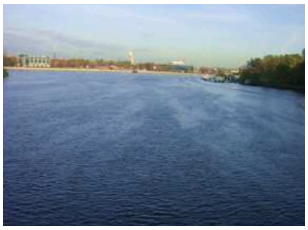
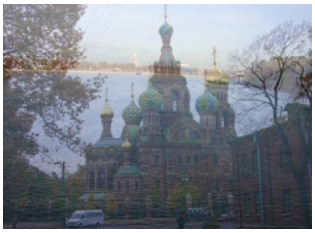
SUMA: mezclar las dos imágenes.

A + B = R

- Para evitar la saturación se usa la media

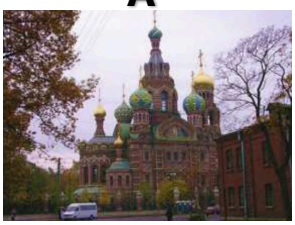
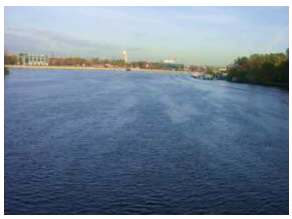
(A + B) / 2 = R

Combinación de imágenes

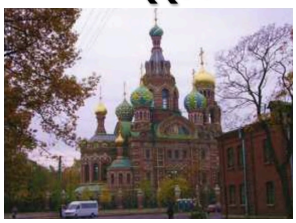
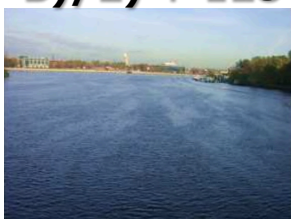
RESTA: obtener diferencia entre imágenes.

A - B = R


- Para evitar la saturación

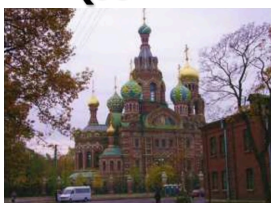
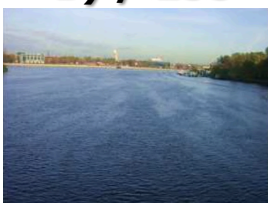
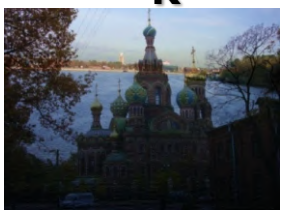
((A - B)/2) + 128 = R


- Si solo queremos conocer la diferencia basta con calcular: R=abs(A-B)

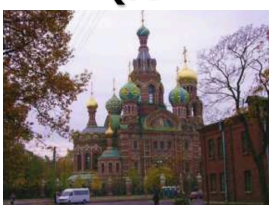
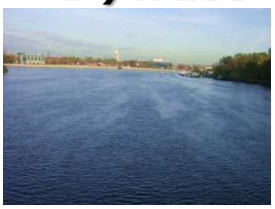
- **PRODUCTO:** efecto de mezcla, similar al de la suma. Realizar escalado dividiendo entre 255.

(A x B) / 255 = R

- **DIVISIÓN:** operación contraria a la multiplicación. Realizar escalado multiplicando por 255.

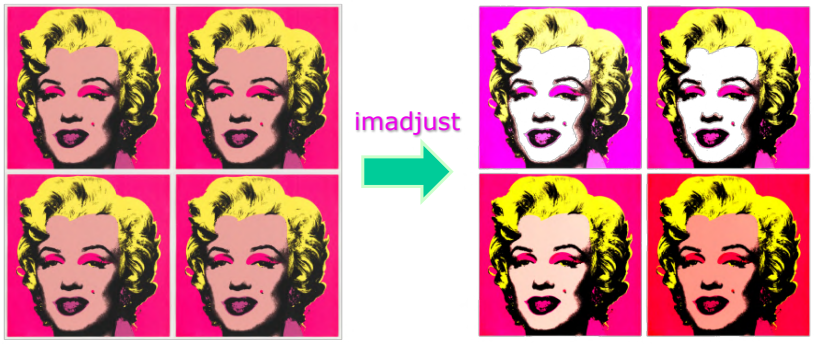
(A / B) x 255 = R


- Combinación de imágenes
  - *Suma*
  - *Resta*
  - *Multiplicación*
  - *División*
- **Montaje de imágenes**
- Efecto Mirror
- Transformaciones geométricas
  - *Trasladar*
  - *Escalar*
  - *Rotar*
  - *Inclinar*

- Mediante la función **montage** realizaremos realizaremos un montaje de las imágenes especificadas.

- Mediante la función **imadjust** podremos resaltar las estructuras de la imagen, obteniendo 4 imágenes con valores de intensidad distintos. Estos valores se encontrarán entre 0 y 1.



- Combinación de imágenes
  - *Suma*
  - *Resta*
  - *Multiplicación*
  - *División*
- Montaje de imágenes
- **Efecto Mirror**
- Transformaciones geométricas
  - *Trasladar*
  - *Escalar*
  - *Rotar*
  - *Inclinar*

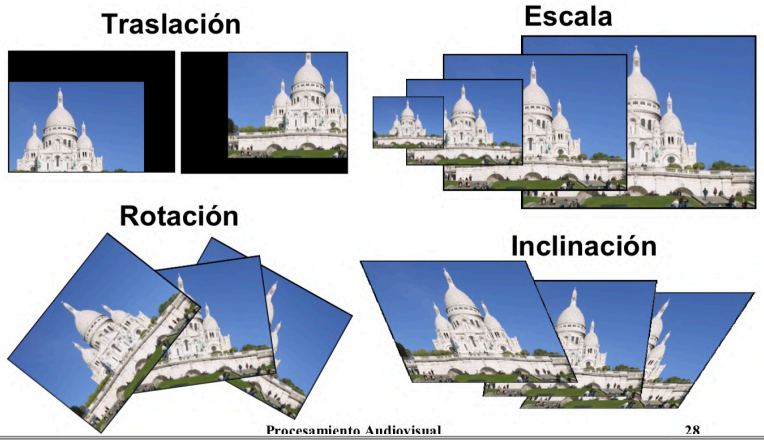
Efecto Mirror

- Consiste en cambiar la simetría de una imagen.
- Esta simetría puede ser vertical u horizontal.
- Se realiza mediante la función flip de Matlab.

- Combinación de imágenes
  - *Suma*
  - *Resta*
  - *Multiplicación*
  - *División*
- Montaje de imágenes
- Efecto Mirror
- **Transformaciones geométricas**
  - *Trasladar*
  - *Escalar*
  - *Rotar*
  - *Inclinar*

Transformación geométrica

- Las transformaciones geométricas modifican la relación espacial entre píxeles. Cada píxel de salida depende de uno de entrada.
- Las transformaciones afines son las más usadas en imágenes digitales 2D.
- Una transformación afín es aquella transformación en la que las coordenadas (x', y') del punto imagen son expresadas linealmente en términos de las del punto original (x, y).



Procesamiento Audiovisual 28

- Los ejercicios en Matlab con extensión .m tendrán el siguiente aspecto, los alumnos tienen que rellenar los huecos con XXX:

Editor – /Users/rocio/Desktop/TFG/rocio/Efectos/efectos.m

efectos.m

```
10
11      %EJERCICIO 1: COMBINAR IMÁGENES:realizar todo tipo de operaciones aritméticas combinando
12      %dos imágenes del mismo tamaño de píxeles
13      %Leemos las dos imágenes que queremos combinar
14 -    imagenA=imread('XXX');
15 -    imagenB=imread('XXX');
16
17      %Realizamos la operación aritmética deseada
18 -    imagen_resultado= XXX;%INCLUIR AQUÍ OPERACIÓN ARITMÉTICA
19
20      %Representamos las dos imágenes combinadas en una
21 -    imshow(XXX);
22
23      %%
24      %EJERCICIO 2: MONTAJE ESTILO WARHOL
25      %Leemos la imagen que queremos representar
26 -    imagen_warhol=imread('XXX');
27
28      %Ajustamos los valores de intensidad de los colores de la imagen (entre 0 y
29      %1)
30 -    J1 = imadjust(imagen_warhol,[XXX XXX]);%CAMBIAR los valores de low y high
31 -    J2 = imadjust(imagen_warhol,[XXX XXX]);%CAMBIAR los valores de low y high
32 -    J3 = imadjust(imagen_warhol,[XXX XXX]);%CAMBIAR los valores de low y high
33 -    J4 = imadjust(imagen_warhol,[XXX XXX]);%CAMBIAR los valores de low y high
34
```

- El guion con los ejercicios a realizar en esta práctica es el siguiente:

CURSO/TALLER TRATÁMIENTO DE IMÁGENES

PRÁCTICA 5 – EFECTOS

Práctica

En esta práctica se va a realizar la aplicación de distintos efectos sobre imágenes. Como ya hemos visto en la parte teórica, son técnicas conocidas y bastante sencillas.

En el código de esta práctica (efectos.m) se comenzará realizando una combinación de imágenes mediante operaciones aritméticas. Esto quiere decir que, mediante la suma, la resta o la división podremos juntar dos imágenes y obtener una imagen combinada como resultado. Para ello habrá que tener en cuenta dos cosas: que el tamaño de las dos imágenes sea el mismo (ejemplo: 400x300 píxeles) y que sean imágenes con claridades distintas, ya que si juntamos dos imágenes muy oscuras o dos muy claras no se podrá observar bien el resultado de la combinación.

En segundo lugar, realizaremos un montaje de 4 imágenes al estilo “Warhol”. Cogeremos una misma imagen y sobre ella aplicaremos distintos valores de intensidad de los colores mediante la función imadjust, obteniendo así 4 imágenes distintas para realizar el montaje deseado.

Otro efecto que aplicaremos será el de “mirror”. Jugaremos con las simetrías de la imagen para obtener una en contraposición de la otra. Esto lo haremos a través de la función flip. Por último, realizaremos distintas transformaciones geométricas como son trasladar una imagen, escalarla, rotarla e inclinarla.

Tratará de que las alumnas jueguen con los distintos efectos de manera que puedan aplicar el que más les guste.

Hacer

Nos descargamos el zip previamente proporcionado denominado Efectos. Extraemos la carpeta contenida en ese zip y pinchamos sobre ella. En su interior podremos encontrar las imágenes correspondientes a esta práctica y el archivo “efectos.m”. Pasos a seguir:

1) Abrimos Matlab  .
2) En la esquina superior izquierda encontraremos una carpeta llamada “Open”  , buscamos la carpeta que hemos extraído y abrimos el archivo “efectos.m”.
3) Realizamos los cambios explicados en el ejercicio.
4) Ejecutamos el archivo pinchando sobre “Run”  .

Ejercicio 1

Combinar imágenes mediante diferentes operaciones aritméticas. Buscaremos en la teoría de esta práctica como aplicar cada una de las operaciones (suma, resta y división). El procedimiento a seguir es el siguiente:

- Leer las dos imágenes. 'montaña.jpg' y 'cascada.png'.
- Escribir la operación aritmética que queremos aplicar.
  (A+B)/2
  ((A-B)/2)+128
  (A/B)x255
- Visualizar la imagen resultado, ejecutar mediante "Run" y observar la imagen combinada.

Haremos esto mismo con cada una de las operaciones aritméticas proporcionadas en la parte de teoría. A la hora de realizar la división tendremos que poner: (A./255)

Ejercicio 2

Realizar un montaje de 4 imágenes al estilo "Warhol". Se leerá una imagen y a través de ella se obtendrán otras 4 con distintos valores de intensidad de colores. Procedimiento:

- Leer la imagen. 'marylin.jpg'
- Cambiaremos los valores que se encuentran entre corchetes [XXX XXX] (con distintos valores en cada línea). Estos valores tienen que estar entre 0 y 1. Ejemplo: [0.1 0.3]
- Visualizar la imagen montaje y ejecutar el archivo mediante "Run".

Probaremos con distintos valores de low y high intentando que las 4 imágenes del montaje queden con colores diferenciados.

Ejercicio 3

En este ejercicio le daremos la vuelta a la imagen creando el efecto "mirror", girándola de izquierda a derecha y de arriba a abajo. Procedimiento:

- Leer la imagen. 'NY.jpg'
- Dar un valor para dim (1 o 2).
- Visualizar la imagen mirror y ejecutar mediante "Run" y observar las imágenes obtenidas.

Ejercicio 4

El objetivo de este ejercicio es realizar la transformación geométrica conocida como traslación. Esta traslación se realiza mediante la función imtranslate que incluye los valores de x e y entre [ ]. El valor x será lo que se traslada la imagen en el eje x y el valor y lo que se traslada la imagen en el eje y. Procedimiento:

- Leer la imagen. 'marylin1.jpg'
- Escribir un valor para x y otro para y. Puede ser positivo o negativo entre 0 y 100. Ejemplo: x = 20;
  y = -10;
- Visualizar la imagen trasladad y ejecutar mediante "Run".

Si el valor de x o de y es negativo, ¿hacia qué dirección se traslada la imagen? ¿Y si es positivo?

Ejercicio 5

En este ejercicio escalaremos la imagen mediante la función imresize. Escalar significa agrandar o reducir el tamaño de la imagen. Devuelve la imagen resultado que es scale veces el tamaño de la imagen original. Para ello haremos lo siguiente:

- Incluir un valor para scale entre 0 y 20. Ejemplo: scale = 0.5;
- Probar con distintos valores de scale.
- Visualizar la imagen escalada y ejecutar mediante "Run".

¿La imagen se hace más grande con un valor menor de 1 para scale o con uno valor mayor que 1?

Ejercicio 6

Rotar la imagen con un ángulo de rotación theta. Cambiar el valor de los grados de este ángulo theta para conseguir distintos resultados. En este ejercicio rotaremos la imagen mediante la función imrotate. Para ello haremos lo siguiente:

- Incluir un valor de theta. Theta puede ser positivo o negativo entre 0 y 360 grados. Ejemplo: theta = 250;
- Probar con distintos valores de theta.
- Visualizar la imagen rotada y ejecutar mediante "Run".

Si theta es positivo, ¿la imagen gira en el sentido de las agujas del reloj o en sentido antihorario?

Ejercicio 7

El objetivo de este último ejercicio es inclinar la imagen. Con maketform crearemos una estructura de transformación espacial. Bastará con cambiar los valores de sh (1) y sh (2) para obtener distintas inclinaciones. Para ello haremos lo siguiente:

- Incluir un valor para sh(1) y otro para sh(2). Valores entre 0 y 10.
  Ejemplo: sh(1) = 1;
  sh(2) = 2;
- Probar con distintos valores.
- Visualizar la imagen inclinada y ejecutar mediante “Run”.

Realizar estos mismos ejercicios pero ahora con imágenes tomadas por la webcam, para ello abrir el fichero efectos_webcam.m.

C Programa 'Campus QSI'

Lunes

9:00 - 9:30	Recepción e Inscripción
9:30 - 9:45	Acto de apertura y bienvenida
9:45 - 10:00	Presentación de la UAM y la EPS
10:00-11:30	Mesa Redonda: El género en las TICs
11:30 - 12:00	Descanso
12:00 - 14:00	Taller elegido por la alumna

Martes

9:30 - 12:00	Taller elegido por la alumna
12:00 - 12:30	Descanso
12:30 - 14:00	Mesa redonda: Estudiar TIC

Miércoles

9:30 - 12:00	Taller elegido por la alumna
12:00 - 12:30	Descanso
12:30 - 14:00	Mesa redonda: Las profesionales TIC

Jueves

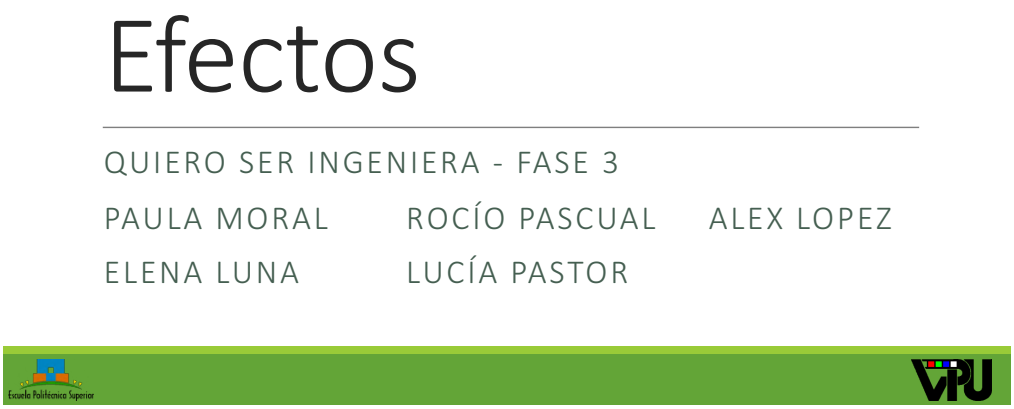
9:30-11:30	Taller elegido por la alumna
11:30 – 12:00	Descanso
12:00 – 14:00	Taller elegido por la alumna

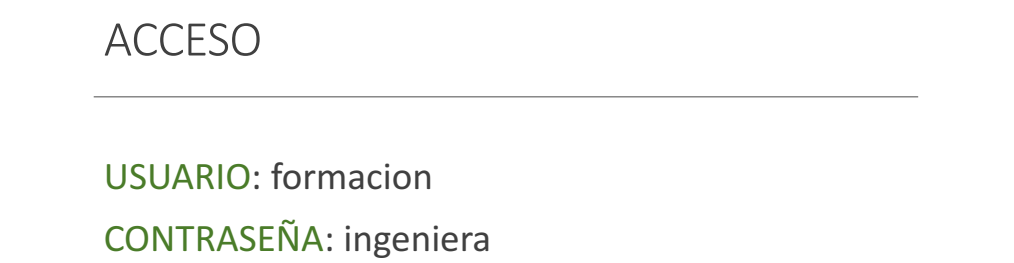
Viernes

9:30-11:30	Taller elegido por la alumna
11:30 – 12:00	Descanso
12:00 – 13:30	Concursos y cierres de cada taller
13:30 – 14:00	Acto de Clausura

D Material adaptado 'Campus QSI'

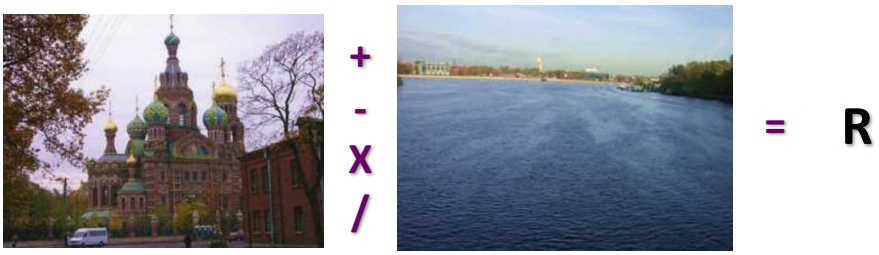
- Las diapositivas utilizadas para la sesión de Efectos en el Campus de 'Quiero ser Ingeniera' son las siguientes, se utilizan como guion y en ellas se puede observar el aspecto que tendrían los scripts .mlx, así como sus resultados:





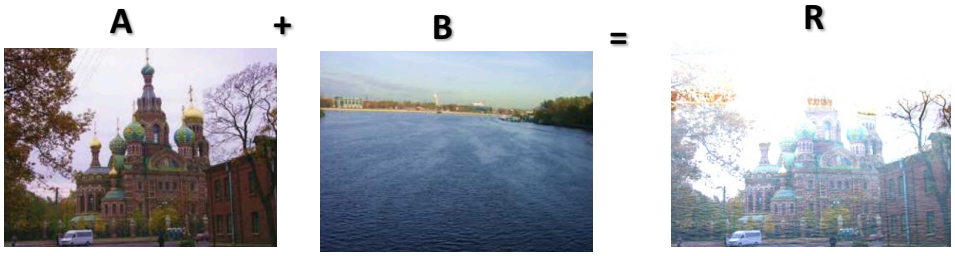
Combinación de imágenes

- Utilizar dos o más imágenes de entrada para producir una imagen de salida.
  - Entrada: imágenes A y B.
  - Salida: imagen R.
- Imágenes con el mismo tamaño.
- Posibles operaciones aritméticas.



Combinación de imágenes

- **SUMA**: mezclar las dos imágenes

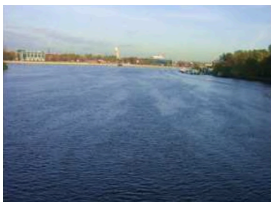


Para evitar la saturación se usa la media

Combinación de imágenes

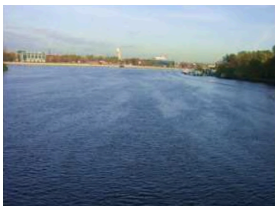
- **RESTA:** obtener diferencia entre imágenes.

A **-** **B** **=** **R**


Para evitar la saturación

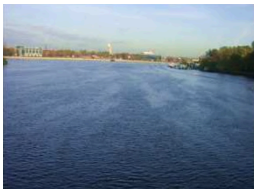
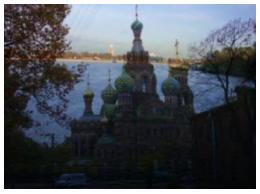
((A **-** **B)/2) + 128** **=** **R**


Combinación de imágenes

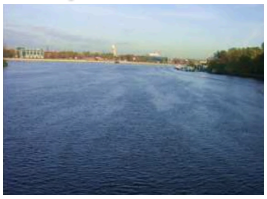
- **PRODUCTO:** efecto de mezcla, similar al de la suma.

(A **x** **B) / 255** **=** **R**

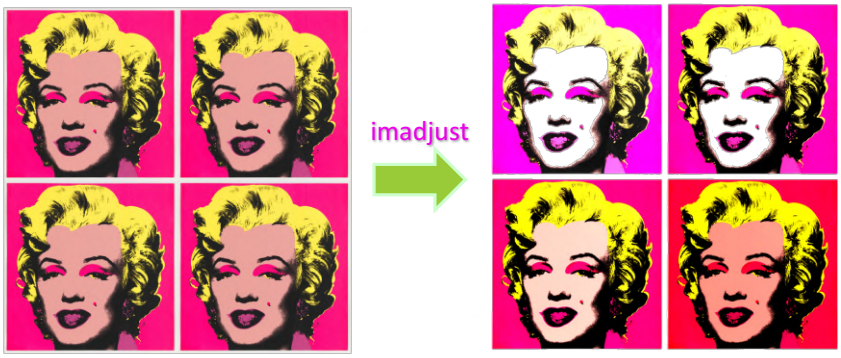
- **DIVISIÓN**: operación contraria a la multiplicación.

(A **/** **B) x 255** **=** **R**


Montaje de imágenes

- Función **montage**: montaje de las imágenes especificadas.
- Función **imadjust:** resaltar las estructuras de la imagen.
- Valores entre 0 y 1.

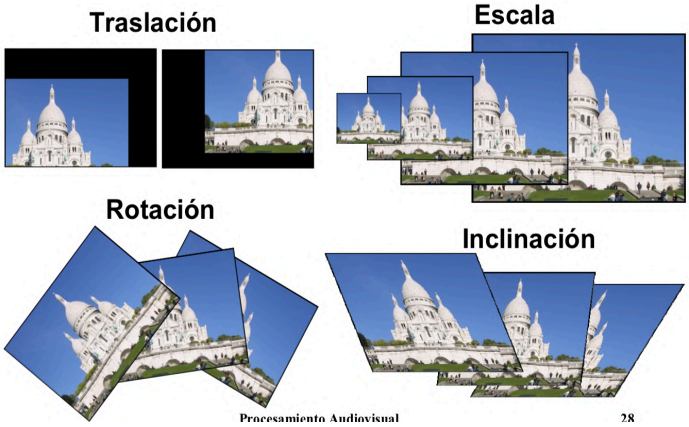


Efecto Mirror

- Consiste en cambiar la simetría de una imagen.
- Simetría vertical u horizontal.
- Se realiza mediante la función flip de Matlab.

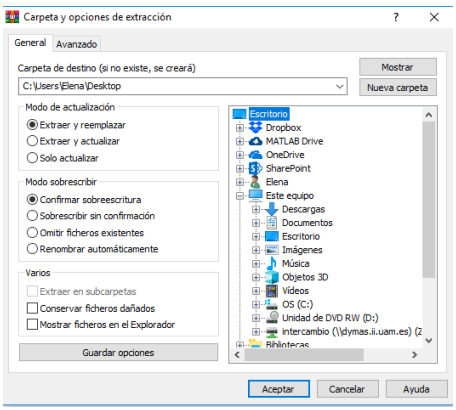
Transformación geométrica

- Modifican la relación espacial entre píxeles.
- Cada píxel de salida depende de uno de entrada.
- Las transformaciones afines son las más usadas en imágenes digitales 2D.



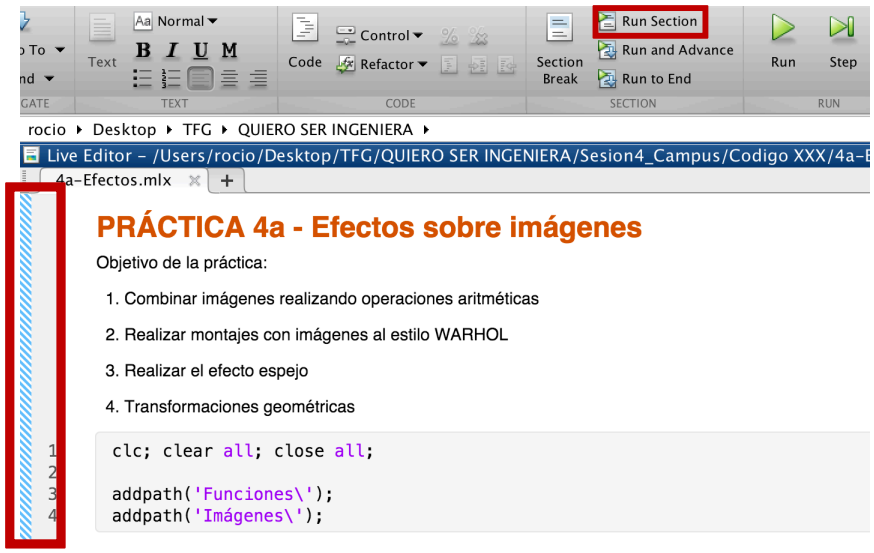
Descargar material

- Google Chrome
  - https://quieroseringenierauam.es/
    - Recursos – Material para las prácticas
      - Material para Sesión 4
- Descomprimir el archivo
  - Carpeta Descargas
    - Extraer Ficheros…
- Carpetas:
  - 4a-Efectos
  - 4b-Efectos con webcam



A partir de aquí las diapositivas sirven como guion para seguir la práctica. En ellas se muestra como vendrán definidos los ejercicios en un script de tipo .mlx.

Efectos



Efectos: Combinación de imágenes

TAREA 1: COMBINAR IMÁGENES, realizar operaciones aritméticas para combinar dos imágenes del mismo tamaño

1.1) Leemos las dos imágenes que queremos combinar A y B.

'montaña.jpg'

'cascada.png'

```
nombre_imagenA='XXX';
imagenA=imread(nombre_imagenA);

nombre_imagenB='XXX';
imagenB=imread(nombre_imagenB);
```

1.2) Escribimos la operación aritmética deseada para obtener la imagen resultado.

Operaciones:

(imagenA+imagenB)/2

((imagenA-imagenB)/2)+128

(imagenA./imagenB)*255

```
imagen_resultado= XXX;
```

Efectos: Combinación de imágenes

1.3) Escribir las dos imágenes A y B y la imagen resultado que queremos visualizar

```
imshow(XXX);
title('Imagen A');

imshow(XXX);
title('Imagen B');

imshow(XXX);
title('Imagen resultado');
```

Efectos: Montaje Warhol

TAREA 2: Montaje estilo Warhol

2.1) Leemos la imagen que queremos representar.

'marilyn.jpg'

```
nombre_imagen='XXX';
imagen_warhol=imread(nombre_imagen);
```

2.2) Ajustamos los valores de intensidad de los colores de la imagen (entre 0 y 1) para obtener 4 imágenes distintas.

[0.1 0.5]

[0.2 0.6]

[0.3 0.7]

[0.3 0.8]

```
imagen1 = imadjust(imagen_warhol,[XXX XXX]);
imagen2 = imadjust(imagen_warhol,[XXX XXX]);
imagen3 = imadjust(imagen_warhol,[XXX XXX]);
imagen4 = imadjust(imagen_warhol,[XXX XXX]);
```

Efectos: Montaje Warhol

2.3) Visualizamos el montaje con las 4 imágenes.

```
figure
montage(XXX);
title('Montaje Warhol');
```

Efectos: Efecto Mirror

TAREA 3: Efecto Mirror

3.1) Leemos la imagen que queremos representar.

'NY.jpg'

```
nombre_imagen='XXX';
imagen_original=imread(nombre_imagen);
```

3.2) Le damos un valor a dim para darle la vuelta a la imagen.

El valor de dimensión debe ser 1 o 2

```
dimension= XXX;
```

Función para dar la vuelta a la imagen.

```
imagen_mirror = girar(imagen_original,dimension);
```

Efectos: Efecto Mirror

3.3) Visualizamos la imagen original y la imagen mirror.

```
imshow(XXX);
title('Imagen original');
imshow(XXX);
title('Imagen mirror');
```

Efectos: Trasladar una imagen

TAREA 4: Trasladar una imagen

4.1) Leemos la imagen que queremos representar.

'marilyn1.jpg'

```
nombre_imagen='XXX';
imagen_original=imread(nombre_imagen);
```

4.2) Escribimos los valores de x e y para trasladar la imagen. Pueden ser valores positivos y negativos entre 0 y 100.

x=20;

y=-10;

```
x= XXX;
y= XXX;
```

Función que traslada la imagen.

```
imagen_trasladada = trasladar(imagen_original,[x,y]);
```

Efectos: Trasladar una imagen

4.3) Visualizamos la imagen original y la imagen trasladada.

```
imshow(XXX);
title('Imagen original');
imshow(XXX);
title('Imagen trasladada');
```

Efectos: Trasladar una imagen

Si x es negativo, ¿hacia donde se traslada la imagen?

Si y es positivo, ¿hacia donde se traslada la imagen?

Efectos: Escalar una imagen

TAREA 5: Escalar una imagen

5.1) Escribir un valor para scale entre 0 y 10

```
escala= XXX;
```

Función para escalar la imagen.

```
imagen_escalada = escalar(imagen_original, escala);
```

5.2) Visualizamos la imagen escalada.

```
imshow(XXX);
title('Imagen escalada');
```


Efectos: Escalar una imagen

¿Con qué valor se obtiene una imagen más grande? ¿Y más pequeña?

Imagen original **Imagen escalada**

Efectos: Rotar una imagen

TAREA 6: Rotar una imagen

6.1) Escribir un valor para el ángulo de rotación entre 0 y 360.

```
angulo= XXX;
```

Función para rotar la imagen.

```
imagen_rotada = rotar(imagen_original,angulo);
```

6.2) Visualizamos la imagen rotada.

```
imshow(XXX);
title('Imagen rotada');
```


Efectos: Rotar una imagen

¿Qué ángulo nos serviría para rotar la imagen hacia la izquierda?

Imagen original

Imagen rotada

Efectos: Inclinar una imagen

TAREA 7: Inclinar una imagen

7.1) Escribir valores para los parámetros 1 y 2 entre 0 y 20. Pueden ser positivos o negativos.

```
parametro(1)= XXX;
parametro(2)= XXX;
```

Obtenemos la imagen inclinada.

```
imagen_inclinada = inclinar(imagen_original,parametro);
```

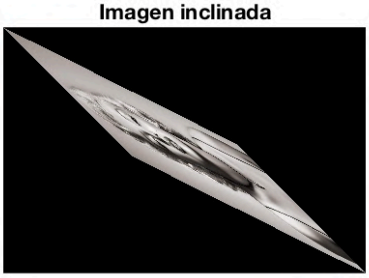
7.2) Visualizamos la imagen inclinada.

```
imshow(XXX);
title('Imagen inclinada');
```


Efectos: Inclinar una imagen

Imagen original

Imagen inclinada



Webcam: Efectos

- Abrir *4b-Efectos_webcam*

TAREA 1: Iniciamos la webcam y tomamos una foto con ella

```
camara = webcam();
preview(camara);
pause(8);
closePreview(camara);
fotoA = snapshot(camara);
clear camara
```

TAREA 2: Iniciamos la webcam y tomamos una segunda foto con ella

```
camara = webcam();
preview(camara);
pause(8);
closePreview(camara);
fotoB = snapshot(camara);
clear camara
```


Webcam: Efectos

TAREA 3: Visualizamos las dos fotos que hemos hecho con la webcam

fotoA

fotoB

```
imshow(XXX);
title('FotoA webcam');

imshow(XXX);
title('FotoB webcam');
```

Realizamos los mismos pasos que para 4a-Efectos.mlx, pero usando las fotos tomadas con la webcam.

E Extensión .mlx

Extensible Markup Language, XML abreviado, describe una clase de objetos de datos llamados documentos XML y describe parcialmente el comportamiento de los programas informáticos que los procesan. XML es un perfil de aplicación o una forma restringida de SGML, el lenguaje de marcado generalizado estándar [ISO 8879]. Por construcción, los documentos XML son documentos SGML conformes.

Los documentos XML se componen de unidades de almacenamiento llamadas entidades, que contienen datos analizados o no analizados. Los datos analizados se componen de caracteres, algunos de los cuales forman datos de caracteres, y algunos de los cuales forman marcas. El marcado codifica una descripción del diseño de almacenamiento y la estructura lógica del documento. XML proporciona un mecanismo para imponer restricciones en el diseño de almacenamiento y la estructura lógica.

[Definición: se utiliza un módulo de software denominado procesador XML para leer documentos XML y proporcionar acceso a su contenido y estructura.] [Definición: se supone que un procesador XML está haciendo su trabajo en nombre de otro módulo, llamado la aplicación.] Esta especificación describe el comportamiento requerido de un procesador XML en términos de cómo debe leer los datos XML y la información que debe proporcionar a la aplicación.